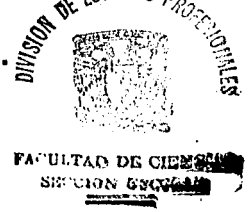
UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CIENCIAS

Detección Automática de Ambigüedades Ortográficas del Español

T E S I S
Que Para Obtener el Título de
M A T E M A T I C O
P r e s e n t a :
Marco Antonio Palomino Zúñiga

México, D.F. 1995



UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

VNIVERSIDAD NACIONAL
AVPNOMA DE
MEXICO

M. en C. Virginia Abrín Batule
Jefe de la División de Estudios Profesionales de la
Facultad de Ciencias
P r e s e n t e

Comunicamos a usted que hemos revisado el trabajo de Tesis:

DETECCION AUTOMATICA DE AMBIGUEDADES ORTOGRAFICAS DEL ESPAÑOL

realizado por MARCO ANTONIO PALOMINO ZUÑIGA

con número de cuenta 9052375-1 **, pasante de la carrera de** MATEMATICO

Dicho trabajo cuenta con nuestro voto aprobatorio.

Atentamente

Director de Tesis Propietario	M.en C. ELISA VISO GUROVICH
Propietario	Dr. MARIO MAGIDIN MATLUK
Propietario	M.en C. JAVIER GARCIA GARCIA
Suplente	M.en C. CLAUDINE COURT MANUEL
Suplente	FISICO VICTOR PEDRO MANTILLA CABALLERO

Consejo Departamental de Matemáticas
M.en C. ALEJANDRO BRAVO MOJICA

FACULTAD
CONSEJO DEPARTAMENTAL
DE
MATEMATICAS

A mi mamá Gloria,

a mi abuelita Toña,

a mi tía abuela Lucha,

a mi bisabuelita Mata;

Dios las bendiga siempre.

MOTIVACIÓN

La corrección de textos debe ser un trabajo tan antiguo como la invención misma del lenguaje escrito. Por supuesto que cada época es distinta y, por tal motivo, los mecanismos involucrados en la corrección de textos a la manera de nuestro tiempo, no se parecen en nada a los utilizados antiguamente, ni se parecerán tampoco a los que en el futuro podrán ser inventados. Actualmente, todas las agencias informativas cuentan con su propia oficina de redacción, y dentro de ella tienen una o más personas especializadas en la corrección de estilo. Lo mismo sucede con cualquier compañía editorial que se respete y, con un poco de imaginación, podemos hacer del trabajo de un corrector, una aventura tan emocionante como la que relata Umberto Eco en *"El péndulo de Foucault"*, donde Jacopo Belbo abandona su flemática y taciturna vida de corrector de estilo, para inmiscuirse en el ambiente de las sociedades secretas y las ciencias ocultas, jugando con el satanismo, la pasión y la perversión, hasta terminar muerto por ahorcamiento tras la lenta y oscilante agonía proporcionada por el histórico péndulo.

En la antigüedad las cosas eran distintas. Cuando en el siglo XIII los manuscritos geográficos hechos por Ptolomeo en griego, circulaban por toda Europa, resultaba que la capacidad para leer el griego, en ese continente, era muy rara, aún entre los instruidos. A pesar de eso, no puede uno imaginarse a Ptolomeo haciendo un pésimo uso del lenguaje, tan sólo porque al fin y al cabo casi nadie lo iba a leer. Así pues, visto de un modo tan simplista como éste, pareciera que la corrección de textos pasó de ser un mero escrúpulo, a ser un trabajo oficialmente legalizado, y, al menos por lo que a la lengua española se refiere, podemos garantizar que el salto del escrúpulo al oficio se ejecutó en 1492, año de la aparición de la primera gramática castellana, escrita por Elio Antonio de Nebrija y dedicada a la reina Isabel la Católica.

Desde los años de Nebrija y hasta nuestros días, la evolución de la lengua española ha sido muy vasta, tanto así, que en cuanto a su estudio y comprensión, lo que antes era empírico, hoy en día es materia de discusiones académicos complejas, que no sólo determinan la correcta ortografía del idioma, sino que también comprometen las reglas del bien decir y escribir. En fin, sea cual sea el enfoque, simplista o rebuscado, la razón de ser de la corrección de textos sigue siendo esencialmente una: garantizar la adecuada transmisión de las ideas por medio de la correcta utilización del lenguaje escrito y, una vez alcanzado este objetivo, mejorar la propiedad, la sencillez y la elegancia de la expresión, a pesar de lo difícil que estos últimos conceptos resultan en su sola definición.

Desafortunadamente, corregir un texto no suele ser tan fácil como quisiéramos. En el caso especial de nuestro idioma, las complicaciones ya de por sí existentes en el manejo de la lengua española, se ven agravadas a causa del deterioro que ésta viene padeciendo desde hace varias décadas. Más de alguno, estará de acuerdo en

MOTIVACIÓN

La corrección de textos debe ser un trabajo tan antiguo como la invención misma del lenguaje escrito. Por supuesto que cada época es distinta y, por tal motivo, los mecanismos involucrados en la corrección de textos a la manera de nuestro tiempo, no se parecen en nada a los utilizados antiguamente, ni se parecerán tampoco a los que en el futuro podrán ser inventados. Actualmente, todas las agencias informativas cuentan con su propia oficina de redacción, y dentro de ella tienen una o más personas especializadas en la corrección de estilo. Lo mismo sucede con cualquier compañía editorial que se respete y, con un poco de imaginación, podemos hacer del trabajo de un corrector, una aventura tan emocionante como la que relata Umberto Eco en *"El péndulo de Foucault"*, donde Jacopo Belbo abandona su flemática y taciturna vida de corrector de estilo, para inmiscuirse en el ambiente de las sociedades secretas y las ciencias ocultas, jugando con el satanismo, la pasión y la perversión, hasta terminar muerto por ahorcamiento tras la lenta y oscilante agonía proporcionada por el histórico péndulo.

En la antigüedad las cosas eran distintas. Cuando en el siglo XIII los manuscritos geográficos hechos por Ptolomeo en griego, circulaban por toda Europa, resultaba que la capacidad para leer el griego, en ese continente, era muy rara, aún entre los instruidos. A pesar de eso, no puede uno imaginarse a Ptolomeo haciendo un pésimo uso del lenguaje, tan sólo porque al fin y al cabo casi nadie lo iba a leer. Así pues, visto de un modo tan simplista como éste, pareciera que la corrección de textos pasó de ser un mero escrúpulo, a ser un trabajo oficialmente legalizado, y, al menos por lo que a la lengua española se refiere, podemos garantizar que el salto del escrúpulo al oficio se ejecutó en 1492, año de la aparición de la primera gramática castellana, escrita por Elio Antonio de Nebrija y dedicada a la reina Isabel la Católica.

Desde los años de Nebrija y hasta nuestros días, la evolución de la lengua española ha sido muy vasta, tanto así, que en cuanto a su estudio y comprensión, lo que antes era empírico, hoy en día es materia de discusiones académicos complejas, que no sólo determinan la correcta ortografía del idioma, sino que también comprometen las reglas del bien decir y escribir. En fin, sea cual sea el enfoque, simplista o rebuscado, la razón de ser de la corrección de textos sigue siendo esencialmente una: garantizar la adecuada transmisión de las ideas por medio de la correcta utilización del lenguaje escrito y, una vez alcanzado este objetivo, mejorar la propiedad, la sencillez y la elegancia de la expresión, a pesar de lo difícil que estos últimos conceptos resultan en su sola definición.

Desafortunadamente, corregir un texto no suele ser tan fácil como quisiéramos. En el caso especial de nuestro idioma, las complicaciones ya de por sí existentes en el manejo de la lengua española, se ven agravadas a causa del deterioro que ésta viene padeciendo desde hace varias décadas. Más de alguno, estará de acuerdo en

aceptar que el mayor culpable de esta situación, es el precario sistema educativo con el que contamos, y que durante los niveles de enseñanza elemental, no parece favorecer mucho la práctica posterior de una apropiada ortografía. También, más de alguno se quejará de las constantes agresiones que nuestro idioma recibe a través de los medios de comunicación, en los cuales, los extranjerismos, los neologismos innecesarios, los vicios de dicción y las deformaciones ortográficas están a la orden del día, listos para ser captados por la mayoría de los hablantes. Sin embargo, tal y como dijimos al principio de este párrafo, estas complicaciones no son las únicas y, de hecho, sólo son agravantes, porque sin necesidad de pensar en ellas, la ortografía del castellano tiene, por sí sola, sus propias dificultades.

Uno de los asuntos que mayores dudas ofrecen en la escritura de un texto en español, es la utilización del acento diacrítico. Según la *Real Academia de la Lengua Española*, el acento diacrítico (del griego diacrós=que distingue) es aquél que sirve para distinguir las funciones que una misma palabra puede asumir en una oración. Así pues, la palabra *el* puede ser un artículo determinado o puede ser un pronombre personal. Cuando sea un artículo no llevará acento, y cuando sea un pronombre sí lo llevará. Para saber si se trata de un artículo, muchas veces no basta con fijarse si está ubicada delante de un sustantivo, pues la misma función la desempeña, por ejemplo, cuando se encuentra delante de un infinitivo sustantivado. La palabra *tú* funge como pronombre cuando está acentuada, y como adjetivo posesivo cuando no lo está. La palabra *sólo* se acentúa únicamente cuando equivale al adverbio *solamente*; cuando significa *sin compañía*, o es un sustantivo, no se acentúa. La palabra *sí*, con acento, trabaja como adverbio de afirmación, sustantivo y variante pronominal, y sin acento funge como conjunción condicional y nombre de una de las notas de la escala musical. La palabra *porqué* se acentúa cuando se usa como sustantivo, y no se acentúa cuando se usa como una conjunción casual. *Por qué* lleva acento en *qué* si se emplea como interrogación directa o indirecta, y no se acentúa en el *que* cuando equivale a *por el que, por lo que, por el cual, por los cuales* o *para que*. Las palabras *éste, ése* y *aquél* llevan acento cuando hacen el oficio de pronombres demostrativos, y no lo llevan cuando son adjetivos, aunque la Real Academia permite que el acento en los pronombres *éste, ése* y *aquél*, vaya de acuerdo con el gusto personal del escritor, siempre y cuando no exista el riesgo de una ambigüedad. A pesar de ello, la mayoría prefiere acentuarlos cuando son pronombres, con el objeto de no confundir al lector. De cualquier modo, esto nos enfrenta ante otro problema: el de aquellas palabras que la Academia acepta con dos formas de acentuación y pronunciación. Ejemplos de ellas son acné-acne, alvéolo-alveolo, cardíaco-cardiaco, reúma-reuma y várice-varice, entre otras.

Como puede observarse, la ortografía castellana tiene en las ambigüedades derivadas de su acentuación, una de sus más serias dificultades. Nótese que un simple vistazo, como el que acabamos de dar a través de las reglas ortográficas, nos ha permitido conocer uno de los tantos problemas del oficio de ser redactor, y si nos fijáramos con mayor detenimiento, podríamos descubrir muchos otros más. Por supuesto que no pretendemos demostrar aquí, que el dominio de una excelente ortografía, es propiedad exclusiva de los académicos, con años y años de estudio en la materia. Antes al contrario, estamos de acuerdo en que cualquier persona de habla hispana, puede llegar a tener una muy buena ortografía, con la ayuda de un poco de práctica; más aún, basta ver que todos aquellos para quienes la literatura resulta ser un pasatiempo apasionante, suelen contar con una magnífica capacidad de expresión, tanto oral como escrita, y además, con una envidiable ortografía, sin tener que haberse dedicado a los estudios especializados. Lo único que sí nos interesa hacer notar, es que el manejo de una buena ortografía no es una habilidad que nos venga dada por instinto, y que, por lo tanto, es indispensable contar con una herramienta que nos permita revisar la ortografía de todos nuestros textos. No podemos dudar de que el escrito hecho por un neófito estará, seguramente, plagado de errores, mientras que el escrito propio de un experto, tendrá sólo unos cuantos, y posiblemente todos ellos serán inconscientes; pero en ambos casos es necesario corregirlos, antes de ser expuestos a un público, por pequeño que éste sea, y convendría poder hacerlo en forma rápida y eficiente, tratando de que la sombra del error humano sea mínima, o de ser posible, nula.

A partir de 1960 el trabajo de la corrección de textos comenzó a automatizarse, a través del empleo de máquinas computadoras. Fue en la década de los sesentas cuando las técnicas de corrección ortográfica automática, y el reconocimiento automático de texto, surgieron como temas de estudio para los investigadores, y como propuestas de trabajo para las compañías de software. En uno y en otro ambiente se inventaron nuevos métodos, mismos que después se mezclarían entre sí hasta dar origen a sistemas razonablemente exitosos.

La experiencia que la mayoría de los actuales usuarios de computadoras, tienen en el campo de la corrección ortográfica automatizada, suele reducirse al conocimiento del corrector de algún paquete comercial de uso generalizado, como por ejemplo, el de la versión 5.1 del procesador de textos de la compañía *WordPerfect*. Después de usar un par de veces dicho corrector, puede parecer que el problema de la corrección ortográfica está concluido, y que ya no tiene mayor futuro, o cuando menos, podría pensarse que cualquier ventaja que pudiera obtenerse en este asunto, estaría relacionada con algún campo que no pertenece, de manera exclusiva, a la corrección ortográfica, digamos, la optimización de recursos de memoria, el perfeccionamiento de las técnicas de almacenamiento de datos, las funciones de acceso aleatorio, etcétera. Sin embargo, resulta sorprendente descubrir, con relativa facilidad, decenas de artículos referentes a este tema, muchos de ellos publicados apenas entre 1990 y 1993, y en los cuales siempre es posible leer, a lo largo de sus páginas, la frase *"quienquiera que considere a la corrección ortográfica un problema resuelto, podría estar menospreciando una serie de sutiles detalles, y otros no tan sutiles, acerca de la naturaleza del problema y del verdadero alcance de los resultados hasta hoy obtenidos"*. No puede negarse que esta situación contribuye a confiar, desde un principio, en que el diseño de un corrector ortográfico es una labor rica en experiencias, útil en aprovechamiento y emocionante en su realización, convirtiéndose entonces en una tarea digna de una investigación seria y, en menor escala, conveniente para el desempeño de un trabajo de tesis de licenciatura.

Obviamente, cuando decimos que la elaboración de un sistema de corrección ortográfica por computadora, sería una buena tesis de licenciatura, nadie pensaría, al menos no en primera instancia, en que ésta sería la tesis de licenciatura de un matemático. Cualquiera pensaría que un aspirante a la licenciatura en matemáticas, es un estudiante que dedica cuatro años de su vida al aprendizaje de los números, a través de la comprensión de una cantidad impresionante de teoremas vinculados entre sí, y que le confieren una habilidad omnisapiente para resolver desde nuestra más insignificante duda relativa a los números pares, hasta el más apabullante de los problemas de un curso introductorio de Análisis Funcional. Comprensión basada, por cierto, en una lógica tajante, dispuesta a juzgar siempre sin prejuicios la validez o la invalidez de un determinado pensamiento, y en la cual, las palabras sólo sirven como un prosaico instrumento para la expresión de las ideas. Aparentemente, para trabajar en un corrector ortográfico se necesita alguien interesado en las palabras, y no en los números, alguien que más que gusto por la lógica sienta gusto por la lengua, la cual no acepta la rigidez de un método inflexible, sino que por el contrario, se maneja con la ligereza de los que cotidianamente la utilizan. No obstante, cuando se pretende hacer ciencia aplicada, no se pueden seguir desde un principio las puntillosas y obtusas cadenas del pensamiento matemático, acompañadas como siempre de su monótona secuencialidad. Primero, tiene uno que iniciar con lo que no aparenta tener ninguna lógica, para luego descubrir que sí la tiene, y poder modelarla. Finalmente, es así como surgió la ciencia, y no hay ni que decirlo. En la Edad Media, mago no era el que no entendía nada y hacía chapuzas con una venda en los ojos, sino el estudioso que intentaba arrancar los secretos ocultos en la materia. Cuántos y cuántos textos absolutamente confiables es posible encontrar, en los que se dice que los físicos positivistas, apenas transponían el umbral de la universidad, iban a desperdigarse en los cotilleos jocosos de las sesiones de espiritismo y de los cenáculos astrológicos, para luego decir que Newton descubrió la Ley de la Gravitación Universal porque creía en la existencia de las fuerzas ocultas. De no haber sido por todo eso, hoy en día no podríamos hablar de ciencia. Por otra parte, algo muy bonito de la corrección ortográfica automatizada, es que apenas uno empieza a conocerla, nos hace sentir invitados al uso de herramientas matemáticas, pues sin ellas todo sería inútil; pero lo más excitante es que dichas herramientas no te las otorga de antemano, te concede crearlas, y eso es lo importante para aquellos que pensamos que no tiene sentido trabajar cuando falta un motivo propio, pues en tal caso es mejor rehacer los trabajos de otros. Ahora bien, una labor como esta, tiene que hacerse dentro de un marco apropiado: el marco de las computadoras, porque sólo ellas nos dejarán realizar cientos de procesos en tiempos breves, y sólo ellas nos permitirán almacenar miles de datos sin que nos estorbe archivarlos. Alguna vez, alguien dijo que el camino más corto entre dos puntos del plano real, se encontraba a lo largo de una curva dibujada sobre el plano complejo, por qué no pensar ahora que algún día alguien dirá, que el contacto más eficiente entre las matemáticas y el lenguaje, se halla a través de un cable de computadora.

En fin, sea como sea, hay que reconocer que el lenguaje, y más precisamente, las palabras, son dueñas de algo muy especial y hermoso, que las pueden hacer capaces de conquistar a cualquiera; cuando conquistan a un hombre con aptitudes excepcionales para la narración, lo convierten en un novelista de altos vuelos, pero cuando

conquistan a un matemático, lo mueven a utilizar toda su capacidad en favor de una expresión no sólo precisa y tajante, sino también refinada y elegante. En estas condiciones, nos concedemos el placer de aprovechar a las matemáticas para diseñar un corrector ortográfico, pues como acabamos de decir, cada quien se enamora de las palabras a su modo, por algo existe aquel viejo proverbio italiano: *"le parolo sonno femini, li fatti maschii"*.

Marco Antonio P. Zúñiga
23/10/1995

INTRODUCCIÓN

Este preámbulo, escrito en un tono más bien coloquial, tiene por objeto preparar el discurso que iniciaremos a partir del capítulo uno. Para ello, necesitamos establecer, claramente, la naturaleza del problema que motiva a este trabajo, pues si nos limitamos a decir que se trata de diseñar un corrector ortográfico por computadora, nuestro planteamiento resultaría en extremo ambiguo, y, por ende, las posteriores explicaciones, quedarían siempre sujetas a la buena interpretación que semejante frase produjera en la imaginación de los lectores. Con este fin, vale la pena comenzar estableciendo la diferencia entre un detector de errores ortográficos, y un corrector ortográfico.

Un **detector ortográfico** es un sistema de cómputo especializado en revelar la presencia de aquellos errores ortográficos que voluntaria, o involuntariamente, son cometidos durante la redacción de un archivo de texto. Para favorecer la comprensión de este concepto, mencionaremos la función que desempeñan los detectores dentro de los lenguajes de programación, porque consideramos que este es un ambiente familiar para los entusiastas de las computadoras.

En todo lenguaje de programación, se define el "vocabulario" como el conjunto de caracteres y palabras con el cual se construyen los programas hechos en ese lenguaje. A los elementos del vocabulario se les acostumbra llamar átomos (*tokens* en inglés), y, por lo regular, se les clasifica en identificadores, constantes, operadores y delimitadores, de acuerdo con su función dentro del lenguaje. Por conveniencia, el conjunto de átomos es seleccionado de modo que el programador pueda expresarse con claridad, y también de modo que el compilador pueda realizar, fácilmente, un reconocimiento eficiente del código durante el proceso conocido con el nombre de análisis lexicográfico. Es justo en ese proceso, donde se centra el trabajo de un detector en un lenguaje de programación, pues en dicho proceso, un buen detector debe, en un solo paso por el código fuente del programa, separarlo en sus átomos constituyentes, clasificarlos sin ambigüedad y advertir la presencia de átomos erróneamente escritos, o sintácticamente incongruentes. Por supuesto, un detector de este tipo, debe aprovechar que el tamaño de su vocabulario es relativamente pequeño, como en todo lenguaje de programación ocurre, y basado en ello, beneficiar a sus usuarios con tiempos de respuesta generosamente breves. Ojalá y que todos los detectores tuvieran siempre esta ventaja, sin embargo, esto casi nunca sucede, debido, entre otras cosas, a que el análisis lexicográfico no es la única aplicación de los detectores de errores, y, de hecho, ni siquiera es la que se puede considerar como la aplicación más socorrida. Comúnmente, los detectores ortográficos son hechos para integrarse en el ambiente de los procesadores de texto, con la intención de revisar documentos en los que un idioma puede ser usado con amplia flexibilidad, a diferencia de los lenguajes de programación, en los que las construcciones gramaticales reflejan siempre una gran rigidez. Por ese motivo, los detectores hechos para procesadores de texto, cuentan con

vocabularios que, en cuanto al tamaño, no tienen limitación alguna, llegando a ser tan abundantes como el de una novela de García Márquez, o tan fértiles y rebuscados como el de un estudio de Noé Jitrik.

La elaboración de un detector ortográfico de uso generalizado para el idioma español, será nuestro primer objetivo en este trabajo. En la actualidad, ya se cuenta con numerosas técnicas encaminadas a realizar esta tarea, y existen bastantes paquetes comerciales de software especializados en ello. La desventaja es que la mayoría de las técnicas desarrolladas, si no es que todas, han sido pensadas para la revisión de textos escritos en inglés, y, por lo tanto, involucran la especial problemática de ese idioma, sin preocuparse por las complicaciones que puedan resultar al momento de aplicarlas en otras lenguas, como, por ejemplo, el español. Aún así, nuestro trabajo estará basado, al menos en principio, en los métodos tradicionales de solución para este tópico, y que con anterioridad han sido estudiados y publicados por diversos investigadores, aunque su efectividad sólo haya sido comprobada para el idioma inglés. De cualquier modo, en lo referente a la detección ortográfica, este trabajo no pretende constituir ninguna innovación significativa, y en el peor de los casos, su única diferencia será la particular implementación que tendrá. Las mayores aportaciones que pensamos realizar en el campo de la automatización ortográfica del español, serán realizadas en lo que consideraremos como nuestro segundo objetivo, un problema de solución más compleja que el de la simple detección.

Un **corrector ortográfico** es un sistema de cómputo especializado en liberar de faltas de ortografía, y posibles defectos de redacción, a un archivo de texto, teniendo como parámetros para determinar los errores, el léxico y las reglas sintácticas, semánticas y gramaticales, propias del lenguaje en el que fue escrito el documento. La versión 5.1 del procesador de textos de la compañía *WordPerfect*, proporciona un buen ejemplo de los sistemas de corrección ortográfica. Una vez que el procesador detecta un error, intenta corregirlo, proporcionándole al usuario una lista de palabras entre las que puede hallarse la corrección, pero dejando que sea el propio usuario quien seleccione el vocablo que, a su juicio, constituye la corrección más pertinente, quedando incluso abierta la posibilidad de rechazar cualquier corrección. Esta característica particular de dejar que sea el usuario quien decida, originó que a este corrector, como a todos los que se le parezcan, se les llame de un modo especial: **correctores ortográficos interactivos**. Existe otro tipo de correctores que realizan su trabajo de manera automática, esto es, sin advertirle al usuario sobre la presencia de errores en su texto, y sin consultarlo tampoco al momento de llevar a cabo las correcciones. A este segundo tipo de correctores, se les nombra **correctores ortográficos automáticos**.

Para una aplicación de procesamiento de texto, los correctores ortográficos interactivos, parecen ser la solución más idónea, no sólo por la conveniente relación que guardan con el usuario, sino también porque la escritura de un texto, no puede quedar constreñida a los infranqueables márgenes impuestos por ningún corrector. Mal que bien, la literatura debe conservar su espontaneidad, y principalmente, la expresividad de sus hablantes, y si muchas veces los autores deciden incluir en sus escritos frases de uso incorrecto, o inclusive errores ortográficos, con la intención de expresar ideas que no podrían manifestarse mejor de otro modo, ningún corrector debe impedírselos. Además, aunque uno pretenda ser un purista de la lengua, en ocasiones es indispensable escribir una que otra palabra en un idioma extranjero. Naturalmente, la excesiva utilización de esta clase de palabras, constituye un síntoma inequívoco de la ignorancia de la propia lengua, pero su exclusión obsesiva rayaría en la imprudencia. Por otro lado, es práctica común, en cualquier lenguaje, incluir algunas frases acuñadas en otros idiomas, por ejemplo en latín, con la finalidad de conferirle elegancia al discurso, o cuando menos para poder citar a los clásicos en su propio idioma. Luego entonces, para aplicaciones de procesamiento de texto, un corrector interactivo resulta ser ideal, pues le concede al usuario la decisión final al respecto del más mínimo detalle. Por su parte, para el mismo tipo de aplicaciones, un corrector automático no funciona, pues ante textos como el de "*El garabato*", de Vicente Leñero, arrancaría errores ortográficos en donde no los hay, y antes de explotar desaforadamente, despedazaría por completo una narrativa como la de Umberto Eco, tan sutilmente acompañada de frases latinas, italianas, alemanas, portuguesas, inglesas y francesas, antiguas y contemporáneas. Debe quedar claro que un corrector automático puede llegar a ser muy eficiente, y que no sirve, únicamente, para distinguir a los correctores que no son automáticos, pero no por ello se le puede exigir que sea más que un simple corrector, de modo que no vamos a convertirlo, por nuestro gusto y conveniencia, en un traductor políglota con sentido común propio.

Muchas de las aplicaciones de la corrección ortográfica por computadora, apenas están por venir, y se

piensa que en ellas el trabajo será, casi exclusivo, para los correctores automáticos. Por ejemplo, en el campo de la comunicación entre los usuarios y las computadoras, el futuro de la corrección ortográfica automática es muy promisorio. Hoy en día, se habla de la innegable existencia próxima de los dispositivos transformadores de material textual en material audible, y también de los nuevos avances en la tecnología del reconocimiento de voz, la cual, eventualmente, nos capacitará para registrar texto a través del habla. En ambos casos, será indispensable la presencia de correctores que revisen, automáticamente, textos que luego tendrán que ser pronunciados sin error, y que corrijan la ortografía de archivos que antes fueron "dictados" por la voz humana. Por supuesto, la carencia de sistemas avanzados de corrección ortográfica automática, repercutirá en la escasez de resultados significativos en estas otras áreas, sobre todo en lo concerniente al desarrollo de aplicaciones en idiomas ajenos al inglés, los cuales han sido minoritariamente estudiados.

Ahora bien, la mayoría de las investigaciones involucradas con la corrección ortográfica, han sido enfocadas hacia la corrección de **palabras aisladas**, es decir, hacia la corrección que para determinar si una palabra es ortográficamente correcta, o incorrecta, sólo se basa en la presencia, o ausencia, de dicha palabra en un diccionario preestablecido, sin tomar en cuenta ninguna información derivada del contexto en el cual aparece. Este método sería excelente para evitar que se cometieran faltas de ortografía tales como *arbol*, *sequia* y *sapato*. Sin embargo, el hecho de que una palabra respete todas las reglas de ortografía, no quiere decir que esté bien escrita, dicho en otros términos, nadie nos garantiza que una palabra sea ortográficamente correcta, tan sólo porque esté presente en un diccionario. Por ejemplo, el vocablo *ala*, escrito así, es ortográficamente impecable, pero aún así podría representar un error ortográfico, digamos en el caso de que un redactor hubiese querido usar la palabra *Alá*, para referirse al nombre con el que los musulmanes llaman a su Dios, y que haya cometido el error de usar *ala*, término con el que se designa a la parte del cuerpo de algunos animales que les sirve para volar, como ocurriría en el caso del siguiente enunciado: "*Mahoma andaba errante por los alrededores de la Meca, dominado por la idea de que le hablaría ala desde el seno de una montaña*". Desde este punto de vista, resulta evidente que la corrección de palabras aisladas, es incapaz de detectar una significativa porción de los errores que se cometen durante la redacción de un documento, pues incluso en el mismo caso del vocablo *ala*, si sólo estamos revisando palabra por palabra, sin considerar que cada palabra forma parte de una oración, y que, por lo tanto, existe un contexto determinado que la engloba, es también probable que estemos descartando la posibilidad de que el redactor hubiese querido escribir la frase *a la*, y que haya incurrido en la falta de juntar las palabras y escribir *ala*, y como esta, otras mil faltas pueden estar haciendo acto de presencia, y nosotros en las nubes. Un método de corrección basado en el contexto, no sólo resolvería este problema, sino que también sería muy útil en la enmienda de aquellos errores que tienen más de una corrección potencial. Ejemplifiquemos la situación diciendo que la palabra *sua* aparece en algún documento que estemos revisando. Obviamente, esta palabra constituye un error ortográfico, pero analizada fuera de contexto, existen muy pocas razones para definir cuál de los siguientes vocablos representa su más óptima corrección: *suba*, *suca*, *suma*, *sura*, *suya*, *sur* y *usa* entre otras. En cambio, si fuéramos capaces de reconocer el contexto dentro del cual fue cometido este error, el número de posibles correcciones disminuiría, llegando, posiblemente, a descartar a todas salvo una, lo cual sería la situación ideal. Este tipo de consideraciones, establece la diferencia entre otras dos clases más de correctores: los **correctores ortográficos independientes**, que corrigen de manera independiente cada vocablo del texto, sin fijarse en la función que éste desempeña dentro del enunciado en el que se halla, y los **correctores ortográficos dependientes de contexto**, que distinguen, de un modo eficiente, las reglas sintácticas y gramaticales del idioma, de modo que puedan revisar no solamente la ortografía de cada palabra, sino también la función que éstas asumen en las oraciones, y con todas esas herramientas, atreverse a resolver, adecuadamente, problemas similares a los que mencionamos con anterioridad. El diseño de correctores de errores independientes, tuvo su origen a principios de los setentas, y se desarrolló con amplitud durante los ochentas. En ese tiempo, los estudiosos del tema tendieron, básicamente, hacia el desarrollo de técnicas eficientes de comparación, para determinar si es que una cadena en especial pertenecía, o no, a una lista predefinida de palabras. El trabajo referente a la corrección basada en contexto, empezó a ser atendido también en la década de los setentas, aunque los artículos de investigación relacionados con este tema, han sido publicados, en su mayoría, a partir de 1980, y comúnmente han pretendido utilizar elementos de procesamiento de lenguaje natural, además de que, a últimas fechas, el interés se ha conducido hacia los recién desarrollados modelos estadísticos del lenguaje. Quizás la mayor desventaja a la que se siempre se han enfrentado los trabajos de corrección ortográfica automatizada, es que los avances obtenidos no parecen satisfacer del todo a las expectativas que se tienen, y la efectividad de los métodos continúa dejando

mucho que desear.

Para realizar un trabajo completo sobre la automatización ortográfica, decidimos que nuestro segundo objetivo fuera, justamente, diseñar un corrector de errores basado en el contexto. Sin embargo, esta es una tarea que se sale de los alcances de un estudio inicial en la materia, como lo es este trabajo, y podríamos dejarlo, más bien, para investigaciones posteriores. No obstante, sí podríamos dedicarnos a lo que creemos que debe de ser la primera característica de un corrector dependiente de contexto del idioma español, a saber, prestar particular atención sobre aquellas palabras que favorecen el surgimiento de **ambigüedades en la expresión escrita**. Con la frase *ambigüedades de la expresión escrita*, designaremos a aquellos vocablos cuya verdadera validez no reside exclusivamente en el hecho de estar bien escritos, sino que depende ampliamente del contexto en el que se utilizan, dado que su significado se altera, notablemente, al quitarles, o ponerles, un acento en alguna de sus vocales, al quitarles, o ponerles, una *h* en su estructura, o al escribirlos con *ll* o con *y*. En este sentido, la presencia de la palabra *el* en un texto en castellano, puede ser considerada una ambigüedad, pues, por un lado, debe de ser necesariamente aceptada por cualquier detector que no examine contextos, pero, por otro lado, eso no quiere decir que esté bien escrita, pues quizá deba de ser acentuada. En una situación similar se encuentran las palabras *ola*, cuyo significado varía al agregarle una *h* al principio, y *halla*, porque podría ser escrita con *ll* o con *y*, según lo que se quiera dar a entender. En ambos casos, un detector no interesado en los contextos, jamás establecería diferencias entre unas y otras, pero un detector de contextos sí lo haría, y por lo tanto, *ola* y *halla* son algunas más de las ambigüedades del idioma español.

Obsérvese pues que dos de los objetivos básicos de este trabajo, serán:

(1) desarrollar un sistema de detección de errores ortográficos, y
(2) desarrollar un dispositivo que le advierta al usuario, sobre la ocurrencia de posibles ambigüedades del español en su texto.

Ahora bien, ¿cómo vamos a conseguir ambas cosas? Bueno, pues empezando por conocer qué es lo que los demás han hecho cuando se han encaminado hacia esfuerzos similares. Esto servirá para que la planeación de los sistemas que hagamos después, tenga buenas bases, además de que nos ayudará a cumplir con lo que será nuestro tercero y último objetivo: la producción de una antología de la automatización ortográfica, desde sus comienzos y hasta nuestros días. De esta forma, el trabajo quedará dividido en dos etapas, una de investigación documental y otra de programación de sistemas. Por razones obvias, la investigación documental tendrá que ocuparnos primero, y no podremos avanzar en el desarrollo de las rutinas de programación, hasta no haber terminado con ella. En cuanto a los idiomas que se tomarán en consideración, la investigación pretenderá revisar todo lo que se ha hecho, principalmente, para el idioma inglés, por ser éste el más estudiado para la automatización ortográfica, pero la parte correspondiente a los objetivos (1) y (2), estará dirigida, en exclusiva, a la lengua española. A lo largo del Capítulo I, estudiaremos a fondo el problema de la detección de errores ortográficos, siempre independientes de contexto, así como las técnicas más usuales en la programación de software especializado en ese fin. Incluiremos también una discusión sobre las variadas complicaciones, y las posibles soluciones, del problema de la elaboración de diccionarios apropiados para proyectos computacionales. El Capítulo II posee una sección dedicada al estudio de los patrones de semejanza más comunes en la comisión de las faltas de ortografía, del mismo modo que se aprovecha para revisar las características genéricas de los defectos en la escritura del idioma, y sus posibles causas. En un cierto sentido, podríamos pensar que el capítulo II constituye un análisis de los errores ortográficos. Para el Capítulo III, preparamos el comienzo de la exposición de las técnicas de corrección independiente, utilizadas en las diferentes metodologías ortográficas hasta la fecha desarrolladas. Por conveniencia a la redacción de nuestro trabajo, y por ventajas para nuestros lectores en la comprensión del material expuesto, decidimos separar la presentación de las técnicas en seis grandes grupos: técnicas de mínima distancia de edición, técnicas de claves similares, técnicas basadas en reglas, técnicas basadas en n-gramas, técnicas probabilísticas y técnicas de redes neuronales. El desarrollo del análisis de estos seis grupos de técnicas de corrección, está distribuido a través de los capítulos III, IV y V, en donde se procuró reunir a los conceptos teóricos con numerosos ejemplos de sistemas programados bajo la óptica de una u otra metodología. Un estudio comparativo sobre la exactitud de los sistemas que han alcanzado un mayor reconocimiento, a causa de las virtudes de su eficiencia,

aparece descrito en el Capítulo VI, con la intención de presentar una especie de medidor de los requerimientos existentes en este campo de estudio, en contra de los resultados hasta el momento obtenidos. El Capítulo VII fue usado para comentar la particular implementación de nuestro detector de errores y ambigüedades, dejando para el Capítulo VIII la declaración de nuestras conclusiones. Hacia el final del trabajo, incluimos tres interesantes apéndices, útiles para aquellos en quienes despertamos la curiosidad por conocer más acerca del ambiente del procesamiento de texto, y de la automatización ortográfica. El Apéndice I contiene una cronología de importantes inventos, descubrimientos, estudios y diseños de programación, involucrados con el tema, además de que en él se concentran todas las referencias de artículos y publicaciones que fueron necesarias para enriquecer nuestra investigación, pero que por no haber tenido acceso directo a ellas, nos está prohibido incluirlas como parte de la bibliografía. El Apéndice II constituye el reporte del análisis de n-gramas que hicimos sobre este mismo escrito, y sobre algunas páginas de un diccionario de la lengua española. Comentarios específicos en cuanto al número total de palabras que usamos, el número de las que fueron distintas, nuestros bigramas más comunes y otros detalles, pueden ser encontrados allí. Finalmente, el Apéndice III sirve como manual del usuario para nuestro sistema de detección ortográfica, en caso de que alguno de los lectores esté dispuesto a emplearlo. Empecemos, pues, con el trabajo por cuyo motivo nos encontramos aquí.

ÍNDICE DE CONTENIDO

CAPÍTULO I
DETECCIÓN ORTOGRÁFICA

Una vez establecido el problema cuya solución nos ocupará a lo largo de las siguientes páginas, procede iniciar la revisión detallada de un bloque importante de información, que nos permitirá ingresar en el ambiente teórico que envuelve a nuestro trabajo. Lo primero será exponer lo que al respecto de la detección, y la corrección ortográfica, se ha desarrollado con éxito hasta el momento. La principal referencia bibliográfica que tendremos en esta tarea, será el artículo titulado *Automatically correcting words in text*, escrito por Karen Kukich, y publicado en diciembre de 1992, por la revista *ACM Computing Surveys*. Comencemos, pues, presentando la situación actual y los avances en lo relativo a la detección de errores ortográficos en texto.

§1. Construcción de diccionarios

Desde el comienzo de las investigaciones relacionadas con los correctores ortográficos, las dos técnicas mayormente utilizadas en la detección de errores, han sido la búsqueda en diccionario y el análisis de n-gramas. La experiencia nos permite afirmar que los especialistas en sistemas de reconocimiento de texto, han encauzado sus esfuerzos hacia el estudio de las técnicas de análisis de n-gramas, mientras que los expertos en corrección ortográfica que no involucra lectura óptica, han preferido las técnicas de búsqueda en diccionario.

La búsqueda en diccionario es una tarea conceptualmente sencilla, pero operativamente compleja. En efecto, la idea que sostiene a los métodos computacionales de búsqueda en diccionario es tan elemental como el sentido común de cualquier persona que redacta un documento: simplemente, escribe sus ideas sobre el papel y cuando le surge alguna duda relativa a la ortografía de una palabra en particular, acude al diccionario. Si descubre que la palabra fue escrita correctamente, entonces no hay ningún problema, la duda desaparece y su texto se conserva igual. Si por el contrario, descubre que la palabra no respeta la ortografía del diccionario, entonces la corrige y punto. Esa es toda la filosofía que hay detrás de las técnicas de búsqueda en diccionario en los sistemas computacionales de corrección ortográfica, la máquina recibe el texto y determina si cada una de las palabras que lo componen se encuentra, o no, en una lista preestablecida de palabras aceptables. En caso de que cierta palabra no se encuentre en dicha lista, la computadora la marca como error ortográfico, con la intención de que el usuario actúe en consecuencia. De este modo, al final de la revisión se obtiene un documento libre de errores, al menos de este tipo de errores. Así pues, las técnicas de búsqueda en diccionario parten de un principio básico muy elemental

y sencillo, aunque no por ello son fáciles de aplicar, y es que su verdadera complejidad viene hasta después, cuando ya no se trata de entenderlas, sino de llevarlas a cabo.

Para poder realizar una búsqueda en un diccionario, lo primero que necesitamos es tener un diccionario, pero ¿cuál es el diccionario más apropiado para un sistema de detección ortográfica? En 1986, los investigadores D. E. Walker y R. A. Amsler intentaron responder a esta pregunta. Su trabajo consistió en comparar un texto común del idioma inglés con un diccionario también del idioma inglés, bastante completo pero general, porque no aceptaron un diccionario de términos técnicos. Como texto escogieron un bloque de ocho millones de palabras del periódico *New York Times*, y como diccionario utilizaron el *Merriam-Webster Seventh Collegiate Dictionary*. Su estudio devolvió como resultado el hecho de que, aproximadamente, dos terceras partes de las palabras del diccionario (61% para ser exactos), no aparecieron en el texto e, inversamente, casi dos terceras partes de las palabras del texto (64%), no estuvieron en el diccionario. Obviamente, nadie esperaba que todas las palabras del texto estuvieran en el diccionario, de hecho, basta pensar que las conjugaciones de los verbos nunca se hallan en un diccionario, a pesar de que cuando uno redacta un documento, siempre utiliza las conjugaciones tanto o más que los infinitivos. Pero aun así, tampoco podíamos pensar que apenas un tercio de las palabras del texto estuvieran en el diccionario. Precisamente con el objeto de determinar la naturaleza de las palabras del *New York Times* que no aparecieron en el *Merriam-Webster Dictionary*, Walker y Amsler tomaron algunas muestras de ellas y las analizaron, concluyendo que un cuarto de esas palabras eran conjugaciones; un cuarto nombres propios; un sexto "palabras separadas por guiones"; un doceavo errores ortográficos y un último cuarto no quedó determinado, aunque en su mayoría se compuso de palabras cuyo ingreso al idioma fue posterior a la publicación del diccionario. El principal problema derivado de esta situación, es que, según Walker y Amsler, muchos de los términos ajenos al diccionario podrían ser frecuentemente utilizados en ciertas aplicaciones de la corrección ortográfica. Por lo tanto, se concluye que aun sin pensar en aquellas aplicaciones que requieren de vocabularios técnicos o especializados, no cualquier diccionario es útil para este trabajo, y peor todavía, los diccionarios convencionales resultan inapropiados.

Con respecto al tópico de la construcción de diccionarios, R. Mitton se abocó a la tarea de preparar una lista de palabras especial para las aplicaciones de corrección ortográfica de uso general. Dicha lista la obtuvo con base en el *Oxford Advanced Learner's Dictionary of Current English*, y contiene un total de 68'000 conjugaciones, derivadas de 38'000 infinitivos, todos ellos incluidos. Posterior a la elaboración de esta lista, G. Sampson decidió evaluarla, revisando con ella un texto de 50'000 palabras escrito en inglés británico, mismo tipo de inglés que Mitton usó en su diccionario. Sampson reportó que la lista de Mitton es sorprendentemente buena, pues sólo 1'477 palabras del texto, digamos un 3.24% del total, no aparecieron en ella. Más aún, 600 de las 1'477 palabras en cuestión, eran nombres propios o adjetivos derivados de nombres propios (por ejemplo, *Carolean*), y algunas más fueron abreviaciones y números escritos con afijos alfabéticos, contratiempo que podría ser corregido con cierta facilidad. El resto de las palabras ajenas al diccionario de Mitton, incluyen palabras extranjeras, variantes morfológicas y especialmente formas negativas (por ejemplo, *uncongenial*, *irreversible*), las cuales siempre han representado problemas para la corrección ortográfica.

Los buenos resultados obtenidos por Mitton en la construcción de diccionarios, nos deberían convencer de utilizar listas de palabras que incluyeran infinitivos y conjugaciones por separado, de modo que siempre pudiéramos contar con unos y con otros durante la detección de errores en un texto. Por otro lado, si el problema de Mitton fueron las variantes morfológicas, ¿por qué no agregarle a la lista todas las posibles variantes morfológicas de las palabras que ya están contenidas en ella? Esta es una pregunta que muchos se han hecho desde que se interesaron en el diseño de diccionarios, pero que no todos se la han respondido de igual manera. A este respecto, conviene recordar que los primeros detectores ortográficos que se inventaron, fueron hechos para equipos de cómputo que disponían de muy poca memoria, lo cual les impidió incluir todas las variantes morfológicas de las palabras que, por cierto, no son dos o tres, sino muchas más, pues incluyen plurales, tiempos pasados, adverbios y nominativos. Por tal motivo, los diccionarios antiguos guardaban únicamente las raíces de las palabras, y si el procedimiento de detección de errores no encontraba una cierta cadena en el diccionario, se pasaba entonces a una rutina de procesamiento morfológico, la cual despojaría a la cadena en cuestión, de los sufijos y los prefijos conocidos del idioma, y posiblemente, la reemplazaría por cadenas alternativas (por ejemplo, *dictionary* por *dictionaries*). Si al final de todo ese trabajo, la cadena seguía siendo desconocida por el detector, la respuesta natural era marcarla como

error ortográfico. En 1988, R. J. Elliot perfeccionó el sistema de procesamiento morfológico de un conocido paquete de corrección ortográfica, haciendo también algunas críticas a los procedimientos hasta ese entonces usados. Entre otras cosas, indicó que los métodos comunes de cancelación de afijos permiten, frecuentemente, la aceptación de palabras erróneas, sobre todo porque tales métodos no procuran observar las reglas gramaticales. Elliot superó esta problemática definiendo clases de equivalencia de afijos, las cuales están determinadas por similitudes ortográficas entre las palabras.

Dentro del mismo orden de ideas, vale la pena agregar que en el campo del procesamiento morfológico, nuestro idioma dispone de una muy interesante investigación académica. A finales del año de 1973, México presenció uno de los más notables avances en el ambiente de la lingüística computacional del castellano, cuando un grupo de estudiosos encabezados por L. F. Lara, R. Ham y M. I. García, consideraron la posibilidad de hacer uso de la computadora en la elaboración del *Diccionario del Español de México*. El inicio de este trabajo, resultó ser bastante poco alentador, como muchas veces le ocurre a los investigadores audaces, pues tuvieron que partir del hecho de que se carecía de un diccionario de la lengua española que sirviera también como base para un diccionario de computadora. Luego entonces, esa habría de ser una de las primeras tareas, tanto para los lingüistas como para los matemáticos involucrados en el asunto. Eventualmente, tuvieron que hacerle frente al problema del reconocimiento de la categoría gramatical de cada uno de los vocablos pertenecientes a un cierto *Corpus*, y con esa finalidad usaron una metodología a la que llamaron el Algoritmo morfológico. Dicha metodología está fundamentada en la eficiencia adquirida en la distinción de la categoría gramatical de una palabra, a través de la comparación de sus caracteres terminales con los morfemas de tiempo, número, persona y modo de los verbos españoles, así como con algunas terminaciones numerales o adverbiales. Explícitamente, en esta investigación, sus autores consideraron que las terminaciones en contra de las que se deberían realizar las evaluaciones eran: *ad, are, mente, irán, endo, cho, so, to, er, séis* y *ó*.

A pesar de los avances logrados en el campo del procesamiento morfológico de palabras, la realidad es que en los últimos años tales avances han quedado enterrados. La razón es que hoy en día, las máquinas que utilizamos cuentan con mucha mayor velocidad computacional, y bastante más memoria, motivo por el cual, la tendencia actual se dirige hacia el almacenamiento de todas las posibles variantes de las palabras, como vocablos separados en el diccionario. Sin embargo, la necesidad del procesamiento morfológico nunca será anulada por completo, pues resulta virtualmente imposible anticipar toda la posible creatividad morfológica de los usuarios, tanto en su expresión oral como en su expresión escrita. De cualquier manera, el incremento de las capacidades de memoria en las computadoras, ha favorecido un cambio de mentalidad entre algunos estudiosos del tema, quienes en vez de preocuparse por evitar diccionarios de muchas palabras, ahora se interesan por organizar la memoria apropiadamente, con el fin de guardar diccionarios de dimensiones monumentales. Ese es, justamente, el sentido del trabajo realizado por J. L. Peterson en 1980, para la distribución adecuada del diccionario en diferentes niveles de memoria. De acuerdo con Peterson, los diccionarios empleados en los detectores ortográficos de los procesadores de texto, deben dividirse en tres niveles de memoria: el primer nivel deberá almacenarse en la memoria *cache*, y consistirá de los pocos cientos de palabras que se utilizan con mayor frecuencia en la práctica del idioma escrito, palabras por las cuales se verifica el 50% de las búsquedas; el segundo nivel habrá de ubicarse en la memoria regular, y se compondrá de los pocos miles de palabras de dominio específico de la aplicación, las cuales representan el 45% de las búsquedas; finalmente, el tercer nivel, almacenado en memoria secundaria, contendrá las decenas de miles de palabras que estén registradas con menor frecuencia de uso, y que sólo motivan el 5% de las búsquedas.

Trabajos como el de Peterson parecen dejar muy clara la conveniencia de usar diccionarios de muchas palabras, pero ésta es una idea que no le convence a muchos. Es evidente que un diccionario pequeño tacharía de errores a bastantes términos válidos, sin embargo, un diccionario muy grande podría incurrir en un número excesivamente alto de aceptaciones falsas, es decir, de errores genuinos que no son detectados porque constituyen una palabra correcta, pero de muy baja frecuencia. Por ejemplo, la presencia de la palabra *agogía* en el enunciado "*La agogía de Sócrates concluyó con las palabras: Critón, le debemos un gallo a Esculapio, no te olvides de pagar esa deuda*", resulta claramente errónea. *Agogía* es el término con el cual se designa al canal del desagüe en las minas, cosa que no tiene nada que ver con el enunciado anterior. Obviamente, la palabra que de seguro intentaron

escribir fue *agonía*, con la que se acostumbra referirse a la lucha postrera de las fuerzas vitales que precede a la muerte. Un diccionario de escaso tamaño hubiera marcado un error al encontrar la palabra agogía en el texto, pero un diccionario demasiado completo la habría aceptado sin mayor problema. Esta es la desventaja de los diccionarios muy grandes.

El mismo J. L. Peterson calculó, en 1986, que aproximadamente la mitad de los errores que resultan al intercambiar el orden de las letras en una palabra válida, producen diferentes palabras aceptables del diccionario, al menos cuando éste contiene un número superior a los 350'000 vocablos. Además, conjeturó que la cantidad de verdaderos errores no detectados podría ser mucho mayor en la práctica, debido a que las palabras de menos letras, las cuales suelen aparecer con mayor frecuencia, presentan una marcada tendencia a transformarse en palabras aceptables cuando es intercambiado el orden de sus letras, o cuando les son agregadas o mutiladas una o dos letras, a pesar de que esto signifique que han sido erróneamente escritas. Con esta idea en mente, Peterson pretendió estimar el porcentaje de errores no detectados en función del tamaño del diccionario. Según sus cálculos, el 2% de los errores pasan desapercibidos ante un diccionario pequeño, el 10% frente a un diccionario de 50'000 palabras, y casi el 16% para un diccionario de 350'000 palabras. Peterson terminó entonces recomendando que el tamaño de los diccionarios se conservase bajo.

Años más tarde, 1989, F. J. Damerau y E. Mays cambiaron la recomendación de Peterson. Analizaron un bloque de más de veintidós millones de palabras, compuesto por texto de varios géneros, y descubrieron que el incremento del número de vocablos en una lista de palabras, capacita al sistema para ubicar errores que no pueden ser detectados por diccionarios con pocas palabras. Damerau y Mays revisaron el archivo de veintidós millones de palabras dos veces, la primera con un diccionario de 50'000 vocablos y la segunda con uno de 60'000. El diccionario de 60'000 palabras detectó 1'348 que no detectó el de 10'000 palabras menos, y, lo más admirable en contra del análisis anterior de Peterson, es que a pesar de ser un diccionario más grande, sólo incurrió en veintitrés aceptaciones falsas extra, es decir, obtuvo un diferencial de error de cincuenta a uno a su favor. Por lo tanto, Damerau y Mays, recomendaron usar diccionarios relativamente grandes.

Como puede observarse, desde el inicio de esta sección y hasta el párrafo anterior, sólo hemos hablado acerca del diseño de un diccionario para aplicaciones de detección ortográfica, lo cual es indispensable, pues como dijimos en un principio, para poder buscar en un diccionario, lo primero que necesitamos es tener un diccionario. Pero también es importante hablar de las técnicas de búsqueda, pues ya que tenemos el diccionario, ahora es necesario aprender a usarlo.

§2. Búsqueda en diccionario

La técnica más común para ganar velocidad de acceso a un diccionario, consiste en el uso de una tabla de dispersión. Este método es particularmente ventajoso para nuestro trabajo de detección ortográfica, porque nos concede la oportunidad de que los tiempos requeridos para la búsqueda en el diccionario, sean independientes del tamaño mismo de la lista de palabras. Cabe hacer notar que la construcción de un diccionario está tan ligada con la búsqueda en el mismo, que no se puede pretender estudiar a la búsqueda aislada de la construcción. El caso de las tablas de dispersión es un claro ejemplo de esta situación, y por consiguiente, nos veremos obligados a comentar primero la construcción de un diccionario especial para las tablas de dispersión, aunque, en realidad, estaremos explicando la metodología de este tipo particular de búsqueda.

A manera de descripción, podemos decir que, en general, las técnicas de dispersión se basan en la utilización de una función h, llamada **función de dispersión**, que transforma palabras en direcciones de un arreglo. Más específicamente, el diccionario es almacenado en una estructura conocida con el nombre de *tabla de dispersión*, la cual está ordenada de acuerdo con la función h, de la forma que se relata a continuación: a cada palabra p, perteneciente al idioma sobre el que se desea trabajar, se le asigna el valor $h(p)$, que representa la posición en la

tabla que habrá de tomar p. La función genera esta dirección a través de operaciones lógicas o aritméticas sencillas, aplicadas sobre toda la palabra p, o sobre alguna parte de p. No conviene hacer aquí una explicación detallada de cada uno de los métodos tradicionales de diseño de las funciones de dispersión, pues, de cualquier modo, la descripción relativa a la función de dispersión que nos importará crear, será dejada para cuando tengamos que comentar la forma en que programamos nuestro sistema de detección de errores.

Quizá el problema que más conflictos produce cuando se trabaja con una tabla de dispersión, es la dificultad de diseñar funciones apropiadas para tal efecto. Usualmente, las funciones de dispersión que conviene usar, o tal vez la única que sirve en un determinado caso, son muy difíciles de deducir a primera vista, y, por lo regular, la experiencia es el único material de consulta del que dispone un programador para poder hacerlas. Apenas hace algunos años, 1992 para ser exactos, E. A. Fox, Q. F. Chen y L. S. Heath publicaron un artículo titulado *A faster algorithm for constructing minimal perfect hash functions*, en donde se describe una manera fácil de preparar funciones muy cercanas al límite ideal que teóricamente se puede alcanzar, sin que eso signifique que siempre vayan a ser cien por ciento óptimas. El problema más característico de las funciones de dispersión, es que, normalmente, no es posible evitar que a dos o más elementos distintos del dominio, la función de dispersión les asigne la misma dirección. Esta situación recibe el nombre de **colisión**, y a los elementos con iguales direcciones se les acostumbra llamar *sinónimos*. En general, para cada función de dispersión que se elabora, es necesario contar con un procedimiento especial de solución de colisiones, procedimiento que en la mayoría de los casos tiene que ver con un método alternativo llamado *encadenamiento*. Dicho método resuelve este problema construyendo, en un espacio ajeno a la tabla, una lista que contiene a todas las palabras de igual dirección. De este modo, si $p_1, p_2, \ldots, p_n$ son palabras tales que $h(p_1)=h(p_2)=\ldots=h(p_n)$ son sus direcciones, entonces, en $h(p_1)$ se encuentra una lista ligada que contiene a las n palabras en cuestión. Alguien podría pensar que si de todos modos vamos a tener que conseguirle espacio a todas las palabras de la lista, sería más apropiado concederle un mayor número de entradas al arreglo para que no hubiese colisiones. Sin embargo, la verdad es que el asunto de las colisiones no se debe precisamente al tamaño de la tabla, sino más bien a que la función de dispersión que la creó no es "perfecta", o para decirlo en términos algebraicos, a que la función de dispersión no es inyectiva casi nunca, razón por la que con todo y una tabla grande, la función no dejará de seguir complicando la tarea. Por lo tanto, antes que pretender evitar las colisiones, lo más acostumbrado es aprender a manejarlas. Nosotros, en su momento, tocaremos este tema con mayor profundidad.

Una vez que se ha programado una tabla de dispersión en un sistema ortográfico, el proceso de detección se inicia desde el momento en que la primera palabra del documento es leída. A cada palabra q, perteneciente al archivo que se está revisando, se le asigna el valor $h(q)$, el cual, de acuerdo con la forma en que se construyó la tabla, debe representar la dirección de q en el arreglo de dispersión. Si la palabra guardada en la dirección $h(q)$ es diferente de q, o bien, si la tabla está vacía en esa dirección, se reporta un error ortográfico. Recordemos que lo normal es que se presenten colisiones, de modo que, luego de haberle asignado a q la dirección $h(q)$, lo normal es que se tenga que recorrer toda la lista contenida en esa dirección, con el fin de hallar a q allí, antes de reportar un posible error ortográfico.

Tal como dijimos antes, los primeros paquetes de detección de la historia, tuvieron que adaptarse a la escasa capacidad de almacenamiento de información que poseían las computadoras de ese entonces. A causa de esta contrariedad, aquellos paquetes no guardaban en la tabla la representación completa en caracteres de todas las palabras, sino que usaban, únicamente, un bit para indicar si es que cada dirección específica de la tabla poseía, o no, un vocablo válido. Este ingenioso truco de programación, parecería muy apropiado incluso para los actuales paquetes de detección, los cuales ya pueden aprovechar equipos con mucha más memoria que los de los primeros años de investigación en la materia. No obstante, este truco, por elegante que sea, acarrea consigo una grave dificultad, debido a que algunas palabras incorrectas tienen, bajo ciertas funciones de dispersión, la misma dirección en el arreglo que algunas palabras correctas. Por ese motivo, los detectores de este tipo dejan pasar como válidas palabras que están mal escritas, para no marcar como errores a otras palabras que sí están bien. Así pues, este método ha resultado ineficiente y, en consecuencia, ha perdido reconocimiento.

Sin temor a equivocarnos, podemos pensar que entre las ventajas que aportan las tablas de dispersión a los

sistemas de detección ortográfica, la principal es la eliminación del gran número de comparaciones que se necesitarían para realizar el mismo trabajo a través de, por ejemplo, búsquedas secuenciales. Quizá otro posible método de búsqueda sería el de búsqueda basada en árboles, aunque su desventaja es que para el adecuado procesamiento ortográfico de un documento, se requieren diccionarios de 25'000 a más de 250'000 palabras, según se dijo anteriormente, y rastrearlas por medio de un árbol, por breve que fuese el documento a revisar, produciría tiempos de respuesta poco aceptables. Puede entonces observarse que las tablas de dispersión constituyen la mejor opción para la búsqueda en diccionario. Con todo y esto, varios estudiosos han desistido de aplicar esta metodología, pues resulta muy complicado determinar una función de dispersión adecuada, en la que pocas palabras, o de ser posible ninguna, tuvieran que ser acomodadas en el mismo lugar de la tabla. En este sentido, las técnicas que han sido desarrolladas con mayor éxito se basan, sobre todo, en la construcción de autómatas de estado finito. Dentro de este campo, A. V. Aho y M. J. Corasick inventaron, en 1975, un método de creación de autómatas de estado finito que representan pequeños conjuntos de términos, llamados palabras clave, y que, posteriormente, habrán de ser comparados con el archivo de texto durante la revisión ortográfica.

Un último método general de búsqueda que conviene mencionar antes de concluir esta sección, porque en ocasiones es utilizado en aplicaciones de detección ortográfica, es el método de los *tries*. Un *trie* es un árbol M-ario cuyos nodos son vectores de M posiciones, todas ellas ocupadas por dígitos o caracteres. Cada uno de los nodos de algún cierto nivel l, representa dentro de esta estructura de datos, el conjunto de todas las palabras que comienza con una cierta sucesión de l caracteres. Knuth asegura que el nombre *trie* fue acuñado por Fredkin en 1960, a causa de la importancia de esta técnica en los mecanismos de recuperación (re*trie*val en inglés) de información, en los cuales Fredkin trabajó con interés.

§3. Análisis de n-gramas

De acuerdo con el diccionario, una *palabra escrita* es la representación gráfica de un sonido, o de un conjunto de sonidos articulados por medio de los cuales se expresan las ideas. Por supuesto, dicha representación debe ser hecha, exclusivamente, con los caracteres correspondientes al alfabeto del idioma al cual pertenece la palabra. Desde el punto de vista de la lingüística, esta definición puede resultar muy conveniente, no en vano es por eso que ésta es la definición del diccionario, pero para los fines computacionales de nuestro trabajo, no es precisamente la definición más apropiada. Por ejemplo, para el desempeño de un detector de errores, la verdad es que no interesa mucho que las palabras represente un sonido, y en un cierto sentido, tampoco nos interesa que las palabras representen una idea, de modo que valdría la pena contar con una definición diferente, que vaya más de acuerdo con nuestros objetivos. Por otro lado, también según el diccionario, una letra es cada uno de los caracteres del alfabeto. Luego entonces, conjuntando las definiciones de palabra y de letra, podríamos preparar una posible segunda definición del término *palabra*, y que diría que una palabra escrita en un cierto idioma, es una sucesión de letras propias del alfabeto de ese idioma. Obviamente, esta última definición de palabra no puede ser tan consistente como la primera, sin antes aclarar que su enunciado recíproco es falso, esto es, que no toda sucesión de letras del alfabeto de un idioma constituye una palabra de ese idioma. Sin esta aclaración en mente, tendríamos que aceptar que cadenas tales como *wgtbm* y *ñlcrs* son palabras del español, tan sólo porque ambas están formadas por letras del alfabeto español. No obstante, una vez hecha esta aclaración, nuestra segunda definición del término palabra, resulta muy práctica, en el sentido de que sólo se basa en el concepto previo de letra, y si bien el número de palabras de una lengua puede llegar a ser muy grande, el número de letras de su alfabeto nunca se excederá de unas cuantas decenas, lo cual implica que es mucho más fácil manejar letras que palabras.

Supongamos que de alguna manera reunimos a todas las posibles sucesiones de hasta veintiún letras formadas con el alfabeto español. Entonces, podremos afirmar que apenas un pequeño subconjunto de ellas conforma el diccionario de la *Real Academia de la Lengua Española*. Para darnos una idea de que tan pequeño sería ese subconjunto, bastaría con considerar que el número total de sucesiones de menos de veintidós letras del alfabeto

español (el cual cuenta, por cierto, con veintisiete letras, descontando a la *ch* y a la *ll*), está dado por la suma

$$S_1=\sum_{i=1}^{21} 27^i=27+27^2+...+27^{21}=\frac{27^{22}-27}{26},$$

pero dado que en nuestra idioma todas las vocales pueden ser acentuadas, y que la *u* puede llevar diéresis, entonces estamos hablando de que, en realidad, el alfabeto español posee treinta y tres caracteres diferentes, de donde el verdadero número de palabras que se pueden formar es

$$S_2=\sum_{i=1}^{21} 33^i=33+33^2+...+33^{21}=\frac{33^{22}-33}{32},$$

mientras que un diccionario común posee apenas 60'000 palabras. Más aún, con todo y las conjugaciones de los verbos y demás variantes morfológicas, la cantidad de vocablos del diccionario difícilmente superaría las dos terceras partes del número S_2. Pero volviendo a considerar a todas las posibles sucesiones del alfabeto español con quince letras o menos, pensemos ahora en la forma de descartar a aquellas que no formen parte del idioma. Con este afán, quizá valdría la pena empezar por observar que muchas de las sucesiones de dos letras no sólo no constituyen palabras de dos letras, sino que tampoco pueden integrarse ni al principio, ni en medio, ni al final de ninguna de las posibles sucesiones de tres o más letras. Por ejemplo, una de las reglas ortográficas del castellano, establece que antes de *l* o *r* siempre se escribe *b*, de manera que las sucesiones *vr* y *vl* no sólo carecen de sentido como palabras de dos letras, sino que tampoco podrían integrarse a ninguna sucesión de tres letras o más. Análogamente, las sucesiones *bb*, *db* y *nb* podríamos proponerlas como ajenas al español, porque después de las consonantes *b*, *d* y *n* siempre se escribe *v*. Si nuestro análisis continúa así, descartaríamos a todas las parejas de letras que no son compatibles con el idioma, y tocaría entonces llevar a cabo el mismo análisis pero con las sucesiones de tres letras. Tendríamos pues, conclusiones del estilo de que sucesiones como *avs*, *ovs* y *suv* están fuera del idioma español, porque las partículas *abs*, *obs* y *sub* se escriben, necesariamente, con *b*. Un análisis semejante para todos los posibles tamaños de sucesiones, nos permitiría determinar el conjunto de sucesiones válidas en el idioma, y de ese modo quedaría más o menos conformada la lista de palabras bien escritas. El problema es que el conjunto de sucesiones válidas podría seguir siendo muy grande, y por ello, muy poco manejable, lo que nos obliga a planear otra estrategia que perfeccione este sistema. Quizá la respuesta sería pensar no sólo en sucesiones de un tamaño específico, sino también en su posición dentro de otras sucesiones de mayor tamaño. Por ejemplo, es apropiado notar que las sucesiones *eba*, *ebe*, *ebi* y *ebo* en caso de formar parte de una palabra no lo podrían hacer al principio de ella, pues los comienzos en *eva*, *eve*, *evi* y *evo* se escriben, por regla general, con *v*. Desafortunadamente, nunca faltan las excepciones que confirman a la regla, y a este respecto podemos decir que los comienzos en *eva*, *eve*, *evi* y *evo* se escriben con *v*, salvo por contadas excepciones tales como *ebanista*, *ebenáceas*, *ebonita* y *eborario*. Por lo tanto, no podremos ser tan tajantes como para cancelar definitivamente a las sucesiones *eba*, *ebe* y *ebo* del principio de todas las palabras castellanas, pero cuando menos sabremos que su frecuencia de aparición al principio de una palabra es mínima, sobretodo si se le compara con la misma frecuencia calculada para las sucesiones *eva*, *eve* y *evo*. Esto último nos sugiere una idea más: manejar nuestro análisis a través de frecuencias, esto es, a través de observaciones estadísticas, con base en las cuales podamos atrevernos a inferir sobre la ortografía del lenguaje, posiblemente sin alcanzar jamás la contundencia, pero bueno, de cualquier modo, las reglas del idioma nunca han sido contundentes. Este es, justamente, el fundamento y el método que se pretende explotar por medio del análisis de n-gramas.

Un n-grama es una sucesión de *n* letras del alfabeto, que puede, ya sea formar parte de una palabra, o bien, ser, por sí sola, una palabra. Conviene aclarar que al referirse a un n-grama, se acepta que n pueda asumir el valor de cualquier número natural, pero lo normal es que *n* sólo represente a alguno de los dígitos 1, 2 o 3. Regularmente, los 1-gramas se conocen por el nombre de monogramas o unigramas, los 2-gramas se acostumbran llamar bigramas o digramas y los 3-gramas trigramas. Por convención, las técnicas de detección de errores a través de n-gramas, realizan su trabajo por medio de la revisión de todos y cada uno de los diferentes n-gramas que

constituyen a todas y cada una de las diferentes palabras del documento sometido a corrección. Aquellas cadenas en las cuales se descubre la presencia de un n-grama inexistente, o cuando menos altamente infrecuente, son identificadas como probables errores ortográficos. Aparentemente, el análisis de n-gramas nos libera de la necesidad de trabajar con diccionario, y por consiguiente, nos concede las ventajas de no tener que lidiar con las palabras hasta conseguir hallarles una adecuada función de dispersión, ni de tener que lidiar con la memoria de la computadora hasta hacer que le quepa el diccionario. Sin embargo, el abandono de las listas de palabras es tan sólo aparente, porque para poder recabar toda la información relativa a un desarrollo con n-gramas, se requiere de un maratónico trabajo previo sobre el diccionario, y es que para saber cuáles son los n-gramas aceptables, cuántos de ellos son frecuentes, cuántos de ellos son infrecuentes, cómo calcularles una adecuada frecuencia estadística, etcétera, etcétera, se requiere de mucho más que un par de referencias al diccionario y a las excepciones de las reglas ortográficas. Aunque si bien es cierto que este método no es apto para programadores flojos, también es cierto que no sólo de flojos está lleno el mundo. Por supuesto, este método nunca dejará de ser muy laborioso, pero no por ello habrá de convertirse en imposible y mucho menos habrá de cancelársele del ambiente de las investigaciones cuando tanto éxito ha tenido en el campo de los dispositivos de reconocimiento óptico de caracteres.

Los sistemas de reconocimiento de texto son capaces, hoy en día, de procesar tres diferentes tipos de documentos: documentos hechos en imprenta, documentos impresos por computadora y documentos escritos a mano (inclusive con letra manuscrita). Con ayuda de los dispositivos de reconocimiento óptico, cualquiera de estos tres tipos de documento puede ser transformado de una hoja de papel en un archivo en disco, para luego ser almacenado y posteriormente utilizado por la máquina. Es común que los dispositivos de reconocimiento trabajen en base al análisis individual de los rasgos propios de cada uno de los caracteres que conforman a las palabras. Así pues, los dispositivos revisan el número de líneas verticales, horizontales, curvas y cruzadas que aparecen en un caracter determinado, con la finalidad de decidir cuál de las posibles letras del alfabeto se ajusta más a ese caracter. En ocasiones, es muy difícil distinguir con exactitud los rasgos que constituyen a una letra, sobretodo cuando la impresión no es de buena calidad, o cuando la letra del usuario es poco legible. Por tal motivo, es común que los dispositivos de reconocimiento cometan errores debidos a la confusión de caracteres con rasgos similares. Luego entonces, no es de extrañarse que un reconocedor de texto registre una *O* en vez de una *D*, una *S* en vez de un *5* o una *t* en vez de una *f*. Obviamente, ningún lector óptico puede ser considerado útil si comete tantos errores de esta naturaleza, lo cual ha conducido a los creadores de esta clase de sistemas, a buscar métodos eficientes que los salven de este aprieto, y es aquí donde se centra la importancia del análisis de n-gramas, pues, al menos para la lectura de textos en inglés, el análisis de n-gramas ha sido ampliamente utilizado. La razón de preferir a los n-gramas para realizar este trabajo, sin tomar en cuenta para ello al resto de las técnicas existentes para detección de errores, tiene mucho que ver con las investigaciones hechas por E. J. Sitar en 1961 y por L. D. Harmon en 1972. A este respecto, Harmon observó que en grandes muestras de texto escrito en inglés, nunca figura el 42% de todos los bigramas posibles, y, más aún, que la sustitución aleatoria de una letra en una palabra inglesa por otra letra del alfabeto inglés, escogida al azar, produce al menos un nuevo bigrama cuya probabilidad de existencia en el idioma es cero el 70% de las ocasiones. Por lo tanto, si sólo se trata de verificar que hayamos leído bien un documento, esto es, si sólo se trata de verificar que lo hayamos leído sin confundir a una letra por otra, los n-gramas parecen proporcionar una excelente herramienta.

La gran desventaja de los n-gramas es que no sirven para corregir, porque como no cuentan con un diccionario de apoyo instalado en su sistema, son incapaces de decidir, y ni tan siquiera de sugerir, cuál es la palabra correcta que debe ser puesta en vez de aquella que contiene un n-grama inaceptable. Sin embargo, volvemos a repetirlo, para la tarea exclusiva de detectar errores, el análisis de n-gramas es un método de excelente utilidad, y por ello ha sido usado ampliamente en las tareas de reconocimiento de texto.

§4. Arreglos n-gramas binarios

Existen diversas formas de almacenar la información necesaria para realizar un trabajo de detección de

errores a través de un análisis de n-gramas. La más simple de todas es la conocida con el nombre de **arreglo bigrama binario**, y que, para una aplicación en lengua inglesa, consistiría de una matriz M de veintiséis renglones por veintiséis columnas, cuyos elementos representarían a todas las posibles combinaciones de dos letras del alfabeto inglés. De este modo, el elemento $M_{1,1}$ estaría asignado al bigrama *aa*, el elemento $M_{1,2}$ al bigrama *ab* y así sucesivamente. En este caso, la información sería codificada de la siguiente manera: si el bigrama constituido por la i-ésima y la j-ésima letras del alfabeto, en ese orden, no aparece ni una sola vez en ninguna de las palabras del idioma, entonces $M_{i,j}$ es igual a cero. Si por el contrario, el bigrama formado por las letras i-ésima y j-ésima, en ese orden, está presente en al menos una de las palabras bien escritas, entonces $M_{i,j}$ vale uno. Con esta explicación, queda claro de donde le viene el nombre de binario a estos arreglos. Naturalmente, para una aplicación en lengua hispana, un arreglo bigrama binario tendría que ser una matriz de treinta y tres renglones por treinta y tres columnas, pues habría que incluir una entrada exclusiva, tanto en renglones como en columnas, para cada una de las veintisiete letras del alfabeto español, cinco entradas especiales para las vocales acentuadas y una única para la letra *u* con diéresis. Convendría entonces que las primeras veintisiete entradas de renglones y columnas correspondieran a las letras del alfabeto, dejando las seis últimas para las vocales acentuadas y para la *ü*. Tendríamos pues, que elementos como el $M_{1,24}$ contendrían un cero, ya que no existe en castellano el bigrama *aw*, mientras que elementos como el $M_{16,14}$ contendrían un uno, porque cientos de vocablos acaban en *on*. También es cierto que una aplicación estricta de la definición de bigrama binario, nos obligaría a asignarle un uno a la entrada $M_{1,1}$, pues, por raro que parezca, sí existe el bigrama *aa* en el español. El problema es que existe en una única palabra que para colmo es muy extraña, a saber, *yaacabó*, que es el término con el que se designa a una cierta ave insectívora de América cuyo canto recuerda las sílabas de su nombre. Por supuesto que en prácticamente todas las apariciones del bigrama *aa*, podremos afirmar que se trata de un error, y, por lo tanto no se perdería nada si decidimos de manera arbitraria que $M_{1,1}$ valga cero.

Resolver el problema del elemento $M_{1,1}$ resultó fácil, fue cuestión de aceptar una simple convención, pero no se debe creer que siempre es tan sencillo, porque existen bigramas poco comunes pero presentes en más de una palabra, y quizá en una palabra muy usada. En este sentido, un bigrama tal como el bigrama *hú* es casi inexistente en el idioma, desde el punto de vista de que casi ninguna palabra lo contiene, aunque aparece en una palabra tan común como lo es *húmedo*. De hecho, el bigrama *hú* sólo aparece en cuatro palabras: *húmedo*, *húmero*, *húmido* y *húngaro*, las cuales no son nada extrañas en un texto en español. De modo que para éste bigrama en especial, cabe decir aquello de que dentro de lo raro es normal, y, por ende, nos coloca en el aprieto de decidir cuál será el valor de $M_{8,22}$, pues aunque casi siempre que aparezca *hú* habrá un error, no nos conviene hacer $M_{8,22}$ igual a cero, porque marcaríamos error en las palabras antes mencionadas. Lo que sí podemos afirmar es que los trigramas *húm* y *hún* existen, y que todos los demás trigramas iniciados con *hú* no existen. Por eso es que un arreglo de trigramas podría ser mucho más útil que un arreglo de bigramas.

Un **arreglo trigrama binario** es igual al arreglo bigrama que ya describimos, con la sola excepción de que el arreglo trigrama cuenta con tres dimensiones, como era de esperarse. Claramente, podríamos seguir la misma línea de pensamiento hasta construir 4-gramas binarios, 5-gramas binarios y demás, pero eso no sería muy conveniente, porque entre más dimensiones tengan los arreglos menos manejables serán, y las búsquedas en un arreglo así cada vez serían más lentas. De cualquier modo, viene al caso decir que a estos arreglos se les acostumbra llamar **no posicionales**, porque sólo manifiestan la existencia de un n-grama, sin por ello indicar su posición en las palabras. Como ya observamos, un arreglo más grande que el correspondiente a un trigrama binario no posicional no es apropiado para nuestros fines, pero la verdad es que se puede ganar mucho si a un trigrama binario lo complementamos con información sobre la posición de los trigramas dentro de las palabras. Por ejemplo, en un **arreglo trigrama binario posicional**, el (i,j,k)-ésimo elemento toma el valor de uno si, y sólo si, existe al menos una palabra en el diccionario con la sucesión de letras *lmn* en las posiciones i,j,k. De este modo, se puede poseer más información sin un consumo adicional de memoria, condición que siempre se anhela aunque no siempre se consigue.

Dos de los estudios que mayores e importantes resultados aprovechables han reportado, al respecto de la efectividad de los arreglos n-gramas binarios posicionales, en la tarea de detección de errores ortográficos para servicio de los dispositivos de reconocimiento de texto, son el de A. R. Hanson en 1976 y el de J. J. Hull y S. N.

Srihari en 1982. Hanson estudió el archivo creado por un lector óptico de caracteres que había procesado un texto hecho a mano y compuesto por palabras de seis letras. El archivo contenía 7'662 palabras en las que existían errores de sustitución de una letra por otra. La mejor detección fue obtenida con ayuda de los arreglos trigramas binarios posicionales, los cuales ubicaron 7'561 de los errores, o sea, el 98% del total, a diferencia de los trigramas binarios no posicionales, que ubicaron una cantidad bastante más pequeña. Por su parte, Hull y Srihari encontraron que la representación en trigramas binarios posicionales de un subdiccionario de 5'000 palabras de 7 letras es capaz de detectar más del 98% de los errores de sustitución. Lamentablemente, nadie continuó el trabajo de estos dos investigadores, y por ello, no se sabe a ciencia cierta como generalizar este resultado para palabras de longitud más corta, ni tampoco para otro tipo de errores que alteran la longitud de las palabras, como por ejemplo, errores de inserción (cuando se le agrega una letra a una palabra), de supresión (cuando se suprime una letra de una palabra) y de *framing* (cuando una letra sencilla es sustituida por una sucesión de dos letras, o viceversa).

Otro paso significativo en el estudio de la efectividad de los arreglos trigramas fue dado por E. M. Zamora, J. J. Pollock y A. Zamora en 1981. Ellos decidieron que el arreglo dejará de ser binario, para que las entradas contuvieran, en vez de ceros y unos, frecuencias estadísticas o probabilidades de aparición de los trigramas en las palabras. Para tal efecto, las estadísticas deben ser recabadas de un texto suficientemente grande, digamos que, por lo menos, de un bloque de un millón de palabras. Con este fin, Pollock y los Zamora analizaron primero una muestra de 50'000 parejas de palabras, en las que cada pareja contenía una palabra bien escrita, acompañada por la misma palabra pero con un error. La idea de ellos era construir un trigrama para las palabras correctas y uno aparte para las palabras incorrectas, con la intención de determinar si es que realmente existía mucha diferencia entre ambos trigramas, lo que garantizaría los resultados posteriores. La conclusión fue que aún cuando el análisis de trigramas es útil para detectar la mayoría de los errores en las palabras mal escritas, no es capaz de distinguir muy bien una palabra válida de una incorrecta. R. Morris y L. L. Cherry también trabajaron sobre este objetivo, e inventaron una técnica alternativa para la utilización de trigramas de frecuencia estadística. Ellos no construyeron su trigrama en base a un texto común del idioma inglés, sino que generaron el arreglo a partir del mismo documento que pretendían revisar. Su razonamiento estuvo basado en su propio descubrimiento de que, para cada autor dado, el número de palabras distintas que utiliza en sus escritos, se incrementa en razón de la raíz cuadrada del número total de palabras que usó. Por lo tanto, si el escritor posee una ortografía medianamente buena, sus errores ortográficos contendrán trigramas peculiares para el documento mismo, y por ello, la mejor fuente de información no estaría dada por los diccionarios ni por los demás textos, sino por el mismo documento que se desea corregir. Nótese que en tal caso, los errores ortográficos serán más bien aquellos cometidos por distracciones durante la escritura, las cuales provocan que los redactores intercambien el orden de las letras, o cometan inserciones o supresiones, y no tanto por verdaderas faltas de ortografía, pues en una persona con realmente mala ortografía, un error ortográfico puede llegar a ser tan frecuente como la más común de las palabras bien escritas. De cualquier forma, Morris y Cherry detectaban los errores según el siguiente procedimiento: preparaban su trigrama de frecuencias basados en el escrito que revisarían, luego, para cada palabra del texto calculaban un número al que le llamaban **índice de peculiaridad**, obtenido como función de las frecuencias de los trigramas que formaban a esa palabra, y, por último, ordenaban a las palabras por índice de peculiaridad, la de índice menor encabezaba la lista, y la de mayor índice iba hasta el final. Su hipótesis de trabajo era que las palabras mal escritas quedarían al principio de la lista, por estar formadas por trigramas infrecuentes, mientras que las palabras bien escritas, formadas por trigramas de frecuencia media y superior, ocuparían las posiciones finales de la lista. El resultado fue todo un éxito: para empezar, sólo se tomaron diez minutos en hacer todo el trabajo con un texto de 108 páginas, y, para confirmar su suposición, veintitrés de las treinta palabras incorrectas quedaron entre las primeras cien palabras de la lista.

Nadie puede negar que Morris y Cherry procedieron de un modo particularmente ingenioso, y que quizá le hayan dado al clavo a muchas de las aplicaciones del análisis de n-gramas. Sin embargo, el problema de la detección de errores ortográficos sigue sin contar con una solución óptima que funcione en todos los casos, y las soluciones convenientes para circunstancias específicas, aunque represente avances significativos, no se pueden considerar problemas cerrados.

Srihari en 1982. Hanson estudió el archivo creado por un lector óptico de caracteres que había procesado un texto hecho a mano y compuesto por palabras de seis letras. El archivo contenía 7'662 palabras en las que existían errores de sustitución de una letra por otra. La mejor detección fue obtenida con ayuda de los arreglos trigramas binarios posicionales, los cuales ubicaron 7'561 de los errores, o sea, el 98% del total, a diferencia de los trigramas binarios no posicionales, que ubicaron una cantidad bastante más pequeña. Por su parte, Hull y Srihari encontraron que la representación en trigramas binarios posicionales de un subdiccionario de 5'000 palabras de 7 letras es capaz de detectar más del 98% de los errores de sustitución. Lamentablemente, nadie continuó el trabajo de estos dos investigadores, y por ello, no se sabe a ciencia cierta como generalizar este resultado para palabras de longitud más corta, ni tampoco para otro tipo de errores que alteran la longitud de las palabras, como por ejemplo, errores de inserción (cuando se le agrega una letra a una palabra), de supresión (cuando se suprime una letra de una palabra) y de *framing* (cuando una letra sencilla es sustituida por una sucesión de dos letras, o viceversa).

Otro paso significativo en el estudio de la efectividad de los arreglos trigramas fue dado por E. M. Zamora, J. J. Pollock y A. Zamora en 1981. Ellos decidieron que el arreglo dejará de ser binario, para que las entradas contuvieran, en vez de ceros y unos, frecuencias estadísticas o probabilidades de aparición de los trigramas en las palabras. Para tal efecto, las estadísticas deben ser recabadas de un texto suficientemente grande, digamos que, por lo menos, de un bloque de un millón de palabras. Con este fin, Pollock y los Zamora analizaron primero una muestra de 50'000 parejas de palabras, en las que cada pareja contenía una palabra bien escrita, acompañada por la misma palabra pero con un error. La idea de ellos era construir un trigrama para las palabras correctas y uno aparte para las palabras incorrectas, con la intención de determinar si es que realmente existía mucha diferencia entre ambos trigramas, lo que garantizaría los resultados posteriores. La conclusión fue que aún cuando el análisis de trigramas es útil para detectar la mayoría de los errores en las palabras mal escritas, no es capaz de distinguir muy bien una palabra válida de una incorrecta. R. Morris y L. L. Cherry también trabajaron sobre este objetivo, e inventaron una técnica alternativa para la utilización de trigramas de frecuencia estadística. Ellos no construyeron su trigrama en base a un texto común del idioma inglés, sino que generaron el arreglo a partir del mismo documento que pretendían revisar. Su razonamiento estuvo basado en su propio descubrimiento de que, para cada autor dado, el número de palabras distintas que utiliza en sus escritos, se incrementa en razón de la raíz cuadrada del número total de palabras que usó. Por lo tanto, si el escritor posee una ortografía medianamente buena, sus errores ortográficos contendrán trigramas peculiares para el documento mismo, y por ello, la mejor fuente de información no estaría dada por los diccionarios ni por los demás textos, sino por el mismo documento que se desea corregir. Nótese que en tal caso, los errores ortográficos serán más bien aquellos cometidos por distracciones durante la escritura, las cuales provocan que los redactores intercambien el orden de las letras, o cometan inserciones o supresiones, y no tanto por verdaderas faltas de ortografía, pues en una persona con realmente mala ortografía, un error ortográfico puede llegar a ser tan frecuente como la más común de las palabras bien escritas. De cualquier forma, Morris y Cherry detectaban los errores según el siguiente procedimiento: preparaban su trigrama de frecuencias basados en el escrito que revisarían, luego, para cada palabra del texto calculaban un número al que le llamaban **índice de peculiaridad**, obtenido como función de las frecuencias de los trigramas que formaban a esa palabra, y, por último, ordenaban a las palabras por índice de peculiaridad, la de índice menor encabezaba la lista, y la de mayor índice iba hasta el final. Su hipótesis de trabajo era que las palabras mal escritas quedarían al principio de la lista, por estar formadas por trigramas infrecuentes, mientras que las palabras bien escritas, formadas por trigramas de frecuencia media y superior, ocuparían las posiciones finales de la lista. El resultado fue todo un éxito: para empezar, sólo se tomaron diez minutos en hacer todo el trabajo con un texto de 108 páginas, y, para confirmar su suposición, veintitrés de las treinta palabras incorrectas quedaron entre las primeras cien palabras de la lista.

Nadie puede negar que Morris y Cherry procedieron de un modo particularmente ingenioso, y que quizá le hayan dado al clavo a muchas de las aplicaciones del análisis de n-gramas. Sin embargo, el problema de la detección de errores ortográficos sigue sin contar con una solución óptima que funcione en todos los casos, y las soluciones convenientes para circunstancias específicas, aunque represente avances significativos, no se pueden considerar problemas cerrados.

CAPÍTULO II
CORRECCIÓN ORTOGRÁFICA INDEPENDIENTE

En las primeras páginas del capítulo anterior, afirmamos que el desarrollo de un sistema de detección ortográfica, era un trabajo cuyo planteamiento sería conceptualmente sencillo, y en efecto, después de exponer el tema a lo largo de unas cuantas páginas, hemos terminado más lejos que nunca de desdecirnos de aquello, quedando plenamente convencidos de que el problema de la detección ortográfica radica en su implementación, más que en su comprensión. Para iniciar este capítulo, no sería malo decir que la corrección ortográfica es un problema totalmente análogo al de la detección, esto es, un problema muy fácil de entender y muy difícil de resolver. La verdad es que esta idea es bastante acertada, al fin y al cabo, cualquiera entiende que para corregir un texto lo único que nos hace falta es saber leer, y claro, disponer de un buen diccionario, con el cual podamos verificar la correcta ortografía de todas las palabras. Por otro lado, también es cierto que cualquiera que sepa de programación, y que haya leído nuestro capítulo sobre detección, sabrá que el sólo hecho de preparar un buen diccionario por computadora, es cuestión de mucho esfuerzo, y que, por lo tanto, la implementación de un corrector de calidad no será trabajo de unas cuantas tardes. Más aún, vale la pena aclarar, desde el primer momento, que el desarrollo de un sistema de corrección ortográfica, será una tarea mucho más compleja que la de la simple detección.

Quizá la mejor forma de plantear el problema del diseño de los correctores ortográficos, sea olvidándonos de las computadoras, y analizando cómo se hacen las cosas cotidianamente. En este sentido, uno de los detalles más apasionantes de la programación, es que no sólo nos obliga a pensar cómo resolver un problema, sino también a pensar cómo pensamos cuando resolvemos los problemas, y eso es justamente lo que haremos en este caso.

Supongamos que tenemos que corregir un texto escrito por nosotros mismos. Entonces, llevamos inicialmente una ventaja: saber siempre qué es lo que se pretendió decir en el escrito, debido a que nosotros mismos fuimos quienes lo hicimos. De este modo, si hallamos una palabra mal escrita el problema es mínimo, porque conocemos perfectamente cuál es la palabra que queríamos escribir, y en cuanto averiguamos su correcta ortografía la podemos corregir. Ahora, supongamos que tratamos de revisar un texto ajeno. Si la persona que lo escribió es un buen escritor, no habrá dificultades, y aunque las haya, será tarea trivial la de corregirlas, porque se entiende bien el texto. Luego entonces, si nosotros mismos hicimos o no hicimos el texto que pretendemos revisar, y si éste tiene buena o mala redacción, tendremos, según el caso, mayor o menor comodidad a la hora de corregirlo, pero de cualquier forma, la más importante de todas las ventajas que podamos llegar a tener, es la de entender el texto. Para mala fortuna nuestra, esta no será la primera ventaja de que dispondremos al momento de la implementación

de un programa corrección ortográfica, antes al contrario, esta será la primera de nuestras desventajas, pues por magnífico que sea un escritor, ninguna computadora va a entender sus escritos.

Supongamos, finalmente, que nos piden leer un documento al cual no se le entiende nada. Entonces, nos colocan en una situación muy embarazosa. A quién no le ha tocado revisar esos escritos en los que pareciera como si las palabras que los forman, hubieran sido recortadas de muy distintas partes, para más tarde ser pegadas por la fuerza en ese documento, y dar origen a un collage desagradable, por lo caótico de su composición, en el que cada palabra tiene que ser vista por separado, y sin aspirar a tener nunca una visión de conjunto que nos satisfaga. Sin que parezca exagerado, la verdad es que así es como lee una computadora, porque ella no entiende, y por lo tanto, las palabras que recibe, y que para nosotros constituyen textos, para ella sólo son sucesiones de letras de diversas longitudes, desencajadas unas de otras a través de caracteres de espacio en blanco, de retornos de carro y de signos de puntuación.

De esta forma, se antoja pensar, al menos en primera instancia, que la corrección ortográfica por computadora sólo se puede de llevar a cabo con la filosofía de una máquina, y no con la de un ser humano que pueda entender. Así pues, pareciera que estamos condenados a revisar siempre palabra por palabra, verificando una sola palabra a la vez, y sin poder considerar a la totalidad del texto, vamos, ni siquiera a la totalidad de un enunciado. Tal y como establecimos en la introducción, los correctores que realizan su trabajo de esta forma, reciben el nombre de correctores ortográficos independientes de contexto, y son muy útiles para limpiar a un documento de aquellas faltas de ortografía que son contundentes, digamos notables a primera vista, porque ese tipo de faltas normalmente genera vocablos inexistentes en el idioma, y que con toda seguridad quedarán al descubierto cuando el texto sea leído palabra por palabra. Sin embargo, como también dijimos en la introducción, los sistemas de corrección ortográfica que no consideren a las palabras como elementos constitutivos de una oración, llevarán siempre por delante un amplio factor de error, por no tomar el cuenta el contexto en el que se desarrolla el discurso de los escritos. En la actualidad, existen algunas técnicas, propias de los correctores ortográficos dependientes de contexto, que minimizan ese factor de error, con resultados bastante decorosos, aunque en algunas otras situaciones dejan mucho que desear, pues todavía están muy lejos de alcanzar la perfección. Por lo demás, vale la pena observar que el incremento en el uso de correctores ortográficos en los procesadores de texto, así como en otras aplicaciones computacionales, ha motivado una considerable reducción en el número de palabras que fuera del contexto y de la gramática, se pueden considerar ortográficamente erradas, de manera que el verdadero reto de hoy en día, es avanzar en el desarrollo de mecanismos de corrección ortográfica dependiente de contexto.

En la primera sección de este capítulo, nos dedicaremos, exclusivamente, a estudiar los patrones de falla más comunes en la comisión de errores ortográficos, así como las faltas ortográficas típicas generadas por dichos errores. Normalmente, se acostumbra decir que en el terreno de las invenciones, todo aquello que se nos ocurra, por extravagante que sea, ya existe, porque seguramente hubo alguien a quien se le ocurrió antes que a nosotros y lo inventó. Esto es justamente lo que ocurre con la búsqueda de semejanzas en los errores ortográficos, y en el conocimiento de los errores ortográficos más comunes. Imagínense que hasta existen listas enteras sobre las más frecuentes faltas de ortografía de los hablantes de una lengua, junto con sus respectivas correcciones. Por increíble que parezca, en verdad existen diccionarios de errores ortográficos: el *Webster's new world misspeller's dictionary*, publicado desde 1983, es uno de ellos. En ese trabajo, se identifican doce diferentes clases de errores ortográficos, y se incluyen discusiones y comentarios sobre los errores cometidos a causa de pronunciaciones indebidas, sobre los errores resultantes de palabras homófonas, disertaciones sobre las causas fonéticas de una ortografía deficiente, y mucho más. Nuestra intención no es la de seguir esa clase de investigación, así que en vez de una lista completa de errores ortográficos, nos encaminaremos hacia estudios más generales. A lo largo de todo el capítulo, explicaremos, por ejemplo, los diferentes tipos de errores ortográficos englobados por dos distintas clasificaciones. Discutiremos también acerca de la cantidad de errores que tienden a presentarse en la escritura de un bloque de texto, y comentaremos las investigaciones hechas por estudiosos que han pretendido analizar la variación del tamaño de las palabras, como resultado de errores ortográficos incluidos en ellas. Todo esto se hará con el fin de determinar si es que las faltas de ortografía pueden ser caracterizados por reglas estadísticas y tendencias probabilísticas. Ese será el objetivo de este primer capítulo dedicado a la corrección ortográfica independiente.

§1. Patrones de semejanza en los errores ortográficos

El estudio de los patrones de error en la ortografía de una lengua, es una técnica consistente en el establecimiento de modelos, o muestras, de las más comunes faltas de ortografía cometidas por los hablantes de esa lengua, durante la práctica de la expresión escrita, con la finalidad de favorecer comparaciones posteriores, o referencias, que contribuyan a la mejor comprensión de la escritura, así como a un apropiado registro de la problemática característica en el aprendizaje y ejercicio de la misma. Si nuestro objetivo de trabajo fuera la enseñanza del idioma, entonces el estudio de los patrones de errores ortográficos sería un excelente comienzo, pero dada la naturaleza de nuestros intereses, solamente los aprovecharemos para planear estrategias útiles de corrección por computadora.

Una de las advertencias más significativas que se derivan de esta técnica, es que no se puede programar, del mismo modo, un corrector para una máquina de escribir, que para un dispositivo de reconocimiento óptico de caracteres, pues un buen corrector para la primera de estas dos tareas, resultaría inadecuado para la segunda, y viceversa. La razón de esta peculiar distinción, es que las faltas de ortografía generadas por un mecanógrafo y por un reconocedor óptico, son de muy distinta índole. Un mecanógrafo transcribe textos a través de un teclado, y por ello es muy normal que aparte de las faltas de ortografía debidas a sus deficiencias en el dominio del lenguaje, cometa errores de coordinación motriz, al deslizar, posiblemente sin intención, sus dedos por una tecla equivocada, o al invertir el orden en algunos de sus teclazos. Por ello, los errores propios de un mecanógrafo tienden a reflejar particularidades relacionadas con el teclado que posee su máquina, así pues, un mecanógrafo suele caer en la falta de sustituir una letra *b* por una letra *n*, o una letra *u* por una letra *j*, debido a que las letras *b* y *n*, así como las letras *u* y *j*, son adyacentes en la mayoría de los teclados, y aún sin deseos de incurrir en faltas, es muy fácil que nuestra coordinación nos traicione, haciendo que el dedo índice oprima la letra *n* en lugar de la letra *b*, o la letra *u* en lugar de la letra *j*. Por otro lado, los errores cometidos por un reconocedor óptico de caracteres, están más bien basados en problemas relativos a la confusión de letras con rasgos similares. Luego, es común que un dispositivo lector sustituya una *D* por una *O*, al no distinguir con precisión entre una y otra. Esto nos permite concluir que al obtener frecuencias de error y distribuciones estadísticas, debemos de estar perfectamente informados del tipo de muestras que las han generado, para más tarde saber cuándo, cómo y dónde usarlas. A este respecto, podemos garantizar que la distribución estadística desprendida de un conjunto de muestras hechas por un mecanógrafo, va a ser muy distinta de la distribución proporcionada por un lector óptico de caracteres. Más aún, un texto copiado de un libro de historia por un mecanógrafo, y un texto de carácter conversacional, tecleado en un servicio de correo electrónico, incluso por el mismo mecanógrafo, pueden generar estadísticas con impresionantes distinciones, tal vez por la notable influencia cognoscitiva y de diálogos casuales involucrada en el segundo texto, misma que está carente en el primero. Por lo tanto, los resultados concluidos a través del análisis de los patrones de semejanza en los errores ortográficos, no sólo no deben ser generalizados a cualquier tipo de captura del texto, sino tampoco a cualquier tipo de texto.

A los errores ortográficos registrados a partir de una revisión de tipo independiente, se les acostumbra clasificar en tres diferentes categorías, estas son:

(1) errores tipográficos,
(2) errores cognoscitivos y
(3) errores fonéticos.

Expliquemos estas tres categorías.

Los errores tipográficos son aquellos producidos por un error de mecanografía, y que casi siempre se derivan de una distracción involuntaria al momento de copiar, o de escribir, un texto. Dicha distracción suele provocar una falla de coordinación que nos impide golpear con los dedos en la tecla apropiada, y que nos conduce a cambiar una letra correcta por una incorrecta, a agregarle una letra a una palabra o al clásico "comernos" una letra. Digamos que escribir *camiób* en vez de *camión*, *ueringa* en vez de *jeringa* y *hol* en vez de *hola*, son ejemplos

de errores tipográficos. Debe observarse que cuando una persona comete un error tipográfico, se asume que no lo hace por ignorancia, sino simplemente por descuido. Esa es, por cierto, la diferencia entre un error tipográfico y uno cognoscitivo. Un error cognoscitivo es una falta de ortografía cuya presencia sí es debida a la ignorancia, y a la deficiencia de conocimientos. En este sentido, el escolar de educación elemental que no ha memorizado bien las reglas ortográficas de las letras *s* y *v*, y que por ello escribe *resivir* en vez de *recibir*, está cometiendo un error cognoscitivo, y que no podrá evitar sino hasta que su preparación le permita hacerlo. Por su parte, los errores fonéticos son una clase especial de errores cognoscitivos, en los cuales el escritor utiliza una sucesión de letras fonéticamente correcta, pero ortográficamente inaceptable, la mayoría de las veces por ignorancia. Por ejemplo, quién no ha oído mencionar que la *h* es muda y que por esa razón más de alguno se confunde, sobre todo durante los primeros años de escritura, y termina escribiendo *abla* por *habla*, o *ilo* por *hilo*. Otra causa común de errores ortográficos de tipo fonético, es aquella provocada por la confusión entre la *b* labial y la *v* labiodental, la cual provoca que se escriba *vurro* por *burro* o *vuitre* por *buitre*. A decir verdad, nuestro idioma es particularmente propicio para los errores fonéticos, pues éstos también pueden derivarse de todos los posibles intercambios de las letras *c*, *s*, *x*, *z*, y las letras *ll* y *y*. En la lengua inglesa, los errores fonéticos tienden a distorsionar la ortografía de los vocablos más que los errores tipográficos, e incluso más que los errores cognoscitivos.

La experiencia y la práctica confirman que es muy complicado asignarle, a cada error en especial, un lugar específico en la clasificación recién explicada, y que, las más de las veces, es muy difícil distinguir entre un error fonético y uno cognoscitivo, y entre uno cognoscitivo y uno tipográfico. Tan sólo por ejemplificar, valdría la pena inmiscuirnos en la bizantina discusión de averiguar si es que alguien escribe *esena*, en vez *escena*, por caer en un error fonético, en uno cognoscitivo o en uno tipográfico, de esos en los que la falla es "comerse" una letra. Para darle solución clara a semejante discusión, tendríamos que entrevistar a quien cometió el error, lo cual sería imposible de hacer con cada una de las faltas de ortografía que viéramos en cualquier parte. Afortunadamente, es innecesario categorizar a los errores para luego poder corregirlos, pues al final de cuentas todos son errores, y todos tendrán que ser tratados de la misma forma. Lo que sí podría ser de mucho beneficio, sería determinar, antes de la programación de un corrector, cuál será la principal clase errores que tendrá que atacar dicho corrector una vez que esté listo, para entonces prestarle particular atención a ese tipo de fallas, e idear la forma de corregirlas del modo más eficiente.

En el año de 1964, F. J. Damerau realizó uno de los primeros, y hasta la fecha más importantes descubrimientos generales, sobre las faltas de ortografía generadas por los seres humanos, y dicho descubrimiento dio origen a una segunda clasificación de los errores ortográficos, misma que ha sido valiosamente aprovechada por muchas de las aplicaciones de este campo de trabajo. Esta segunda clasificación no obedece a las causas productoras de los errores, como lo hace la clasificación anterior, sino a la forma adquirida por un vocablo que ha sido erróneamente escrito, en comparación con el mismo vocablo cuando ha sido bien escrito, razón por la que esta segunda clasificación podría llamarse clasificación morfológica.

La clasificación morfológica de los errores ortográficos se fundamenta en la observación de que, aproximadamente, el ochenta por ciento del total de las palabras erróneamente escritas, contiene una, y sólo una, de las siguientes cuatro instancias de error: inserción, supresión, sustitución o transposición de letras. Los errores ortográficos que pueden identificarse por medio de una sola de estas cuatro instancias, reciben el nombre de errores ortográficos sencillos, mientras que aquellos que contienen más de una de tales instancias, se conocen como errores ortográficos de calidad múltiple. El nombre de cada una de las instancias expresa con bastante claridad su significado, y por ello no hay mucho que decir al respecto. Obviamente, un error por inserción consiste en agregarle una letra a una palabra bien escrita, un error por supresión se basa en eliminar una letra de una palabra bien escrita, un error por sustitución consiste en sustituir una letra de una palabra bien escrita por otra, y, finalmente, un error por transposición es aquel en el cual se intercambia el orden de dos letras de una palabra bien escrita.

Si bien el trabajo de Damerau ha sido siempre un pilar fundamental de numerosas aplicaciones, también ha sido rebatido varias veces. Por ejemplo, en el año de 1984, J. J. Pollock y A. Zamora encontraron que apenas un 6% de un total de 50'000 palabras erróneamente escritas, eran errores ortográficos de calidad múltiple, en tanto que para Damerau estos errores ocupaban hasta un 20% del total. Tiempo después, 1987, Mitton halló que el 3%

de los errores ortográficos presentes en un texto de 170'016 palabras, contenidas en ensayos escritos a mano, poseían errores ortográficos múltiples. La conclusión obtenida al cotejar estos dos resultados es, nuevamente, que no se debe de generalizar ninguna conjetura, dado que las diferentes clases de textos, y los diferentes métodos de captura que existen, hacen variar mucho los errores de un escrito a otro, convirtiendo en inútil cualquier generalización. Como ya hemos dicho antes, los errores ortográficos generados por los dispositivos de reconocimiento óptico de caracteres, no siguen los patrones propios de los errores generados por humanos, y por eso, los trabajos de Pollock y Zamora no se pueden juzgar bajo las mismas condiciones que los de Damerau. Relativo al tema de los lectores ópticos, podríamos esperar, por motivos que deberían de ser obvios a estas alturas, que el grueso de los errores debidos al empleo de los dispositivos de reconocimiento óptico de caracteres, sean sustituciones. Con todo y eso, no pueden hacerse muy contundentes afirmaciones relativas a esta suposición. Según reportaron en 1991 M. A. Jones, G. A. Story y B. W. Ballard, los errores producidos por los reconocedores de caracteres, varían ampliamente de un reconocedor a otro, así como de una calidad de texto a otra. Jones también descubrió que una significativa porción de los errores de un dispositivo óptico, no forman parte de los que se llaman errores uno a uno. Un error uno a uno es aquel en el cual una letra, o una sucesión de letras, es sustituida por otra letra, o por otra sucesión de letras; digamos, cuando se escribe la cadena *ri* por la letra *n*, o la letra *m* por la cadena *iii*, se está incurriendo en un error uno a uno. Como quiera que sea, los errores uno a uno siempre serán de más fácil corrección que los demás, porque en ellos se sabe, de antemano, que el problema es único, en tanto que en los otros errores, el sólo hecho de identificar el problema, podría ser muy complejo. En los últimos años, el aumento en el uso de dispositivos de reconocimiento óptico, y las crecientes innovaciones que ha habido en ese campo, han favorecido la investigación a ese respecto. Así pues, los nuevos investigadores han establecido nuevas clasificaciones especiales para el reconocimiento óptico. Posiblemente, la más conocida de estas nuevas clasificaciones es la J. R. Rhyne y C. G. Wolf, propuesta en 1993 y según la cual, los errores de reconocimiento caben, necesariamente, en alguna de las siguientes cuatro categorías:

(1) sustituciones,
(2) ilegibilidad, es decir, cuando el caracter leído no se asemeja a ninguno de los caracteres del alfabeto,
(3) inserciones o supresiones y
(4) errores ortográficos uno a uno.

§2. Longitud de las palabras

La longitud de una palabra se define como el número de letras que contiene esa palabra. Luego, se dice que *casa* tiene longitud cuatro, que *radiotelecomunicación* tiene longitud veintiuno y que *cardenal* tiene longitud ocho. Una observación importante hecha sobre la longitud de las palabras de habla inglesa, establece que, normalmente, la diferencia de longitudes entre una palabra bien escrita, y la misma palabra pero con un error ortográfico sencillo, es menor o igual que dos. Esto ha sugerido a muchos investigadores, particularmente a aquellos interesados en la corrección ortográfica especial para los dispositivos de reconocimiento óptico, la estrategia mejorar sus sistemas con una partición de sus diccionarios, para luego generar subdiccionarios de búsqueda alternativa. Tal partición debe ser hecha pensando en la longitud de las palabras como parámetro de referencia. Teóricamente, este método reduce el tiempo de búsqueda y selección de vocablos durante la ejecución del corrector. Desafortunadamente, para que esta conjetura acerca de la longitud de las faltas de ortografía y de sus correspondientes correcciones, fuese en realidad una conveniente herramienta de la automatización ortográfica, necesitaríamos que todas las palabras bien escritas del idioma, tuvieran una longitud específica, para que entonces sí pudiéramos corregir a través de una observación directa de las longitudes. Sin embargo, esto no es posible, y por ello existen muy pocos datos concretos sobre la relación entre los errores ortográficos en las palabras, y las longitudes que estas tienen. Uno de esos pocos datos existentes, es que los errores cometidos en las palabras de poca longitud, llamadas **palabras cortas**, son más difíciles de corregir que los de las **palabras largas**, o de elevada longitud. En 1935, con intereses completamente ajenos a las computadoras, G. K. Zipf determinó que las palabras

cortas aparecen con mayor frecuencia que las largas, y, posteriormente, en 1973, T. K. Landauer y L. A. Streeter, ahora sí con fines computacionales, realizaron un estudio empírico del que concluyeron que las palabras de alta frecuencia, o sea las palabras cortas, tienden a tener más errores ortográficos sencillos que las palabras de baja frecuencia, por lo que en ellas se tiene que invertir un mayor tiempo de corrección. Por su parte, Pollock y Zamora afirman que las palabras cortas causan muchas complicaciones para los correctores, independientemente de su frecuencia de aparición. En sus propias palabras, "*aunque los errores ortográficos en palabras de tres o cuatro caracteres [palabras cortas], constituyen apenas el 9.2% del total de errores ortográficos, este tipo de errores genera un 42% de correcciones equivocadas*". Además, para variar, la proporción de errores que se presentan en palabras cortas depende, en mucho, de la aplicación sobre la cual se trabaja. En 1983, E. J. Yannakoudakis y D. Fawthrop estudiaron 1'377 errores ortográficos encontrados en obras literarias, y esto les permitió saber que la frecuencia de aparición de los errores en las palabras cortas era muy escasa, tan solo un 1.5%. Sin embargo, en 1990 K. Kukich analizó 2'000 errores pertenecientes a textos conversacionales, y determinó que más del 63% de los errores se ubican en palabras de longitud dos, tres y cuatro. Semejantes diferencias de información, enfatizan la necesidad de determinar las verdaderas características de los errores ortográficos de la aplicación que pretendemos trabajar, antes de diseñar e implementar un sistema de corrección.

§3. Errores en la primera posición

Generalmente se piensa que son muy pocos los errores ortográficos que se presentan en la primera letra de una palabra, pero muy pocos investigadores se han aplicado a la tarea de verificar esta suposición. Al igual que en las últimas secciones, las mejores referencias de que disponemos, son los trabajos de Pollock y Zamora, Yannakoudakis y Fawthrop, y Mitton y Kukich. Los primeros, Pollock y Zamora, encontraron que el 3.3% de sus 50'000 palabras mal escritas, contenía un error en su primera letra; luego, Yannakoudakis al lado de Fawthrop, observaron que en un bloque de 568 palabras erradas, 1.4% de ellas contiene una falta de ortografía en la primera posición. Mitton, por su parte, halló que el 7% de todas las faltas de ortografía que el estudió, contaban con un problema en la primera posición, y, finalmente, Kukich, en el más reciente de todos los estudios sobre el tema, concluyó que el 15% de un total de 40'000 palabras extraídas de textos conversacionales, poseía errores en su primera posición. De este modo, se concluye que los errores en la posición inicial de una palabra son muy escasos, y, por ende, tal vez podríamos pensar en hacer caso omiso de ellos. En primera instancia, esto parece ser una idea descabellada, por cuanto implicaría sumergirnos en una hermosa fantasía en la que los hombres no podríamos incurrir en faltas de ortografía al principio de ninguna palabra, lo cual podría ser un perfecto argumento para un cuento de Sir Lewis Carroll, pero que también sería, o al menos parecería ser, un razonamiento por demás inconsciente para un programador de sistemas. No obstante, descartar la posibilidad de que exista un error ortográfico en la primera posición, nos permitiría, entre otras cosas, dividir a nuestro diccionario en un total de veintisiete subconjuntos, uno por cada letra del alfabeto, en donde guardaríamos a las palabras sin contar con su primer caracter. Esta situación, según dicen los expertos, reduciría ampliamente los tiempos de búsqueda en el diccionario, aunque, por supuesto, no se puede tener todo en la vida, y en la computación menos que en la vida, por lo tanto, la ganancia en el tiempo de respuesta, se compensaría con una reducción en la exactitud de la misma, pues en el supuesto caso de que, contrario a nuestra suposición, el error sí se hubiera presentado en la primera posición, su corrección no se encontraría nunca en el subconjunto en el que la buscamos, lo cual nos conduciría a abortar, sin éxito, un proceso, por causa de una inadecuada suposición. Una observación importante, es que la estrategia de olvidarse de la primera letra de cada palabra, así como las investigaciones que la han motivado, sólo han sido probadas en la lengua inglesa. Desconocemos con exactitud qué ocurriría en el caso de ser aplicada al español, aunque asumimos que los resultados no serían de alcances tan favorables como en el inglés, porque en el español existen muchas más causas que posibilitan la comisión de errores ortográficos en las primeras posiciones. Por ejemplo, en el español existen muchas palabras que comienzan con una *h*, y que fonéticamente parecieran iniciar con alguna vocal, o con otra letra, situación que motiva cualquier cantidad de errores en textos escritos por personas cuyo conocimiento del idioma es deficiente. Una situación similar se presenta con las confusiones que en la primera posición causan las palabras iniciadas con *s*, *x* y *z*. Por último, en el español, a diferencia del inglés, existe el acento

escrito, y si bien no son demasiadas las palabras acentuadas en la primera letra, si proporcionan una rica fuente de posibles errores ortográficos.

§4. Relación entre los errores ortográficos y el teclado

En secciones anteriores hemos hecho comentarios referentes a los errores comúnmente cometidos por los mecanógrafos. Por ejemplo, hemos dicho que suelen incurrir en errores tipográficos, y que sus errores muchas veces revelan cuáles son las teclas adyacentes en sus teclados, pues al adquirir velocidad en la transcripción, una pequeña pérdida de la concentración, los hace presionar dos teclas juntas al mismo tiempo, o simplemente teclear una por otra. Cualquiera diría que la manera de evitar los problemas de los mecanógrafos, sería procesar el texto de alguna otra forma, porque se antoja imposible imaginar siquiera, que los mecanógrafos fueran perfectos, sin que eso signifique que menospreciemos su trabajo. Los dispositivos de reconocimiento óptico de caracteres, se apuntaban como la más fiable de las soluciones a este problema, aunque su drástica metodología consista en erradicar los problemas de los mecanógrafos, erradicando a los propios mecanógrafos. Verdaderamente pensaríamos que son la solución más apropiada, pues al trabajar en un cien por ciento con las máquinas, se elimina toda posibilidad de aparición del factor de error humano. No obstante, ya también hemos discutido que el reconocimiento óptico no constituye la panacea universal, puesto que así como un mecanógrafo se equivoca, un dispositivo de reconocimiento se confunde, al menos cuando la calidad del texto no satisface determinadas condiciones.

Si volvemos otra vez al asunto de los mecanógrafos, podríamos pensar que tanta insistencia en analizar todos los gajes de su oficio, no podría ser gratuita, y que algún provecho habría de tener. Lo malo es que hasta hace poco tiempo, casi en ninguna técnica de corrección ortográfica se habían intentado explotar, directamente, las tendencias probabilísticas que surgen como consecuencia de estudiar los problemas de un mecanógrafo con el teclado. Fue hasta el año de 1983, cuando D. R. Gentner, J. Grudin, S. Larochelle, D. A. Norman y D. E. Rumelhart, miembros todos del *Grupo de investigación mecanográfica LNR*, se olvidaron de desarrollar un sistema de corrección ortográfica, y en vez de eso se dedicaron a trabajar en un modelo de simulación de mecanografía por computadora. Como parte de este trabajo, el grupo *LNR* preparó un extenso análisis sobre los errores de mecanografía cometidos por seis expertos mecanógrafos, y por ocho mecanógrafos novatos, durante la transcripción de un total de 60'000 caracteres de texto, extraídos de un conjunto de artículos de revistas. Las diferencias existentes entre la velocidad de los expertos, comparada con la de los novicios, así como los tipos de errores cometidos por unos y por otros, fueron muy grandes. Entre otras cosas, se observó que la mayoría de los errores de los expertos fueron inserciones, en los que suelen incurrir por golpear dos letras adyacentes del teclado al mismo tiempo, a causa de un exceso en la velocidad de transcripción. Por su parte, los errores de los novatos casi siempre fueron sustituciones, aunque mientras que el porcentaje de error de un experto varió entre el 0.4% y el 1.9%, los novatos promediaron, entre todos, un 3.2%.

Como siguiente paso en la investigación, el equipo *LNR* construyó una matriz de confusión, a partir de los datos recopilados sobre los errores de sustitución cometidos por la totalidad de los mecanógrafos. Una matriz de confusión es una matriz cuadrada, cuyos renglones y columnas están etiquetados con los caracteres del teclado. El *ij*-ésimo elemento de la matriz, representa el número de veces en que la letra *i* fue erróneamente tecleada, cuando la letra *j* debió haber sido escrita. El total de errores de sustitución de los mecanógrafos fue de 3'800, y todos ellos fueron registrados en la matriz de confusión del grupo *LNR*. Los resultados corroboraron las observaciones desde antes señaladas. En efecto, se comprobó que la mayoría de los errores de sustitución, el 58% para ser exactos, involucró la presencia de letras adyacentes en el teclado. Los investigadores también descubrieron una estrecha relación entre los errores de sustitución y la frecuencia de aparición que las letras tienen en el idioma. A este respecto, resultó que la tendencia de los mecanógrafos se inclina a sustituir una letra de alta frecuencia en el idioma, por una letra de baja frecuencia, pero que es "vecina" suya en el teclado. Más tarde, y luego de tan detallada investigación, el equipo *LNR* se aventuró a modelar un sistema de mecanografía por computadora, el cual estuviera capacitado para generar la misma clase de errores que un ser humano.

Reiteramos que prácticamente ninguna investigación ha sido hecha con la intención de estudiar mayormente la relación entre el teclado, el factor humano y los errores, pero el *Grupo de investigación mecanográfica LNR* parece haber marcado la pauta.

§5. Porcentajes de error

El objetivo de este párrafo es dar a conocer la razón entre el número de palabras escritas, o capturadas de alguna otra manera, y la cantidad de errores ortográficos cometidos en ellas. Es imposible generalizar estos resultados, porque como era de esperarse, los datos varían mucho dependiendo del tipo de texto y del modo en que éste fue capturado. Por tal motivo, se ha convenido en clasificar los porcentajes de errores ortográficos de acuerdo con los siguientes parámetros:

(1) Tamaño del texto sobre el cual se llevó a cabo el estudio.

(2) Tipo de texto y modo de captura del mismo, es decir, referencias explícitas que indiquen, por ejemplo, si el estudio se practicó sobre un texto escrito a mano y capturado a través de reconocedor óptico de caracteres, o si se procesó a través de una interfase basada en pluma. En caso de tratarse de un texto mecanografiado, se debe indicar si fue de tipo documental o de tipo conversacional.

(3) Fecha del estudio, porque las investigaciones más recientes, sobre todo en el campo de los procesadores de texto, revelan porcentajes de error cada vez más bajos que en años pasados, debido a la actual disponibilidad de correctores ortográficos en algunas aplicaciones computacionales.

(4) Tipo de corrección ortográfica realizada durante la investigación, misma que, según sabemos, sólo puede ser de tipo independiente o de tipo dependiente de contexto.

Así pues, considerando estos cuatro incisos, podríamos clasificar al estudio mecanográfico del grupo *LNR*, del cual hablamos en la sección anterior, como un estudio hecho sobre 60'000 caracteres de texto mecanografiado, realizado en 1983, y en el que no se distinguió entre errores independientes y errores dependientes de contexto. En este trabajo, se concluyó que el porcentaje de error de un mecanógrafo experto es del 1%, mientras que el de un mecanógrafo novato es del 3.2%.

Más sobre el porcentaje de comisión de errores ortográficos, fue estudiado en otros dos interesantísimos trabajos posteriores al del equipo *LNR*, los cuales midieron, tal vez sin querer, la capacidad ortográfica de las personas con deficiencias auditivas. Ambos fueron hechos con base en textos conversacionales mecanografiados por usuarios del *Servicio de telecomunicaciones para sordos*. El primero de ellos fue realizado por Y. C. Tsao en 1990, y el segundo por K. Kukich en 1992. Tsao analizó 130'000 palabras, y encontró que el porcentaje de errores reconocidos por medio de un sistema de corrección ortográfica independiente, es del 5%. El texto de Kukich contenía 40'000 palabras, y el porcentaje de error calculado fue del 6%.

En 1984, Pollock y Zamora examinaron veinticinco millones de palabras de un nutrido bloque de texto, con ayuda de un lote de máquinas lectoras de bases de datos. Una corrección ortográfica independiente, les permitió ubicar 50'000 palabras mal escritas, de donde concluyeron que el porcentaje de error fue del 0.2%, lo cual es bastante bajo. Por supuesto que a todos nos gustaría ser perfectos, y por eso es que el sólo hecho de hallarnos un

error, por insignificante que éste sea, nos produce un mal sabor de boca, pero tenemos que reconocer que un porcentaje de error del 0.2% es magnífico, en el sentido de que es tan bajo que a más de uno nos gustaría tenerlo. En efecto, si tuviéramos dos únicas faltas de ortografía en cada mil palabras que escribiéramos, nos podríamos considerar unos expertos en la escritura del idioma, y en caso de que todos fuéramos así, hasta es posible que los sistemas de corrección ortográfica nunca le hubieran interesado a nadie, más que como un lujo, o como una tarea del curso de Estructuras de Datos. Sin embargo, lo más impresionante es que hoy en día es posible encontrar estudios que han revelado porcentajes de error todavía más bajos. Por ejemplo, en 1991, K. W. Church y W. A. Gale trabajaron sobre un bloque de texto noticioso publicado por la agencia *Associated Press (A.P.)*. En ese estudio, definieron un *typo* como una palabra rechazada por el programa *Spell* de *UNIX*. Al finalizar su trabajo de investigación, concluyeron que *A.P.* genera, aproximadamente, un millón de palabras y 500 *typos* por semana, lo cual equivale a decir que su porcentaje de error es del orden del 0.05% (¡increíble!). Naturalmente, *A.P.* dispone de correctores ortográficos computarizados.

De la misma forma en que existen investigaciones especializadas en mecanógrafos, en publicaciones periodísticas y en servicios de telecomunicaciones, también existen estudios encaminados a revisar la ortografía de individuos en nivel escolar, a quienes no se les puede considerar expertos mecanógrafos, ni tampoco expertos lingüistas. A este respecto, A. M. Wing y A. D. Baddeley trabajaron sobre un conjunto de textos formado por cuarenta ensayos hechos a mano por estudiantes del *Colegio de Cambridge*. Los ensayos totalizaban más de 80'000 palabras y, aproximadamente, 1'185 errores ortográficos. A lo largo de este trabajo, los investigadores emplearon correctores ortográficos independientes y dependientes de contexto. Un análisis superficial de su material, les permitió descubrir que al menos un 30% de los errores ortográficos cometidos por los estudiantes, sólo podría ser detectado por un corrector dependiente de contexto. Con similares intenciones, R. Mitton examinó, en 1987, novecientos veinticinco ensayos escritos por estudiantes de preparatoria, los cuales sumaban un total de 170'016 palabras y 4'128 errores. Mitton también trabajó con correctores ortográficos independientes y dependientes de contexto, pero sus conclusiones fueron más contundentes que las de Wing y Baddeley, pues, según su estudio, el 40% de los errores ortográficos sólo pueden detectarse con correctores dependientes de contexto. Al igual que Wing y Baddeley, Mitton determinó que los errores que no se pueden detectar por los correctores simples, involucran, usualmente, sustituciones de palabras correctas por palabras erróneas, aunque, por supuesto, en este caso las palabras erróneas no lo eran por contener faltas de ortografía, sino por poseer connotaciones gramaticales ajenas a las que se pretendían. Por ejemplo, las sustitución del verbo *arrive* por su pretérito *arrived*, la del pronombre personal *he* por el también pronombre *her*, o la incorrecta división de la palabra *myself* por la frase *my self*. Un punto más en favor de la corrección ortográfica dependiente de contexto, lo proporciona un estudio hecho por J. L. Peterson en 1986, y del cual ya habíamos mencionado algo antes. Peterson descubrió que al menos el 16% de los errores ortográficos simples, convierten a una palabra correcta en otra palabra también correcta, pero diferente de la original; de modo que ese 16% pasará inadvertido para los detectores ortográficos independientes. Este es el tipo de afirmaciones que claman por la necesidad de mayor atención a la investigación de métodos de corrección ortográfica dependiente de contexto.

Antes de finalizar este resumen relativo a los porcentajes de aparición de errores en distintas clases de textos, consideramos apropiado citar un último comentario sobre los dispositivos de reconocimiento óptico de caracteres, que quizás más que un fragmento de investigación documental, parecerá un desmantelamiento de un truco comercial de mal gusto. El asunto es que la mayoría de los fabricantes de equipo para reconocimiento óptico, asegura que sus artefactos proporcionan un porcentaje de error mínimo, del orden del 1% o cuando mucho del 2%. Sin embargo, una reciente investigación hecha por Santos y Jones, probó que, en la práctica, los dispositivos de reconocimiento óptico de caracteres, tienen porcentajes de error de hasta el 7% o el 16%, porque en aplicaciones reales, estos dispositivos no cuentan con las condiciones ideales proporcionadas por el ambiente del laboratorio.

§6. Reglas heurísticas y tendencias probabilísticas

Finalmente, hemos llegado a la parte concluyente de este capítulo, cuyo objetivo es, precisamente,

descubrir, a través de la observación tendencias probabilísticas, un cierto patrón de comportamiento en la comisión de errores ortográficos, con la finalidad de diseñar eficientes heurísticas de corrección por computadora. Aquí mencionaremos las condiciones y los resultados de tres de los estudios más sobresalientes en este campo, el primero de ellos realizado por los investigadores Yannakoudakis y Fawthrop, el segundo, por el equipo de Pollock y Zamora, y el tercero por Kernighan. Una poderosa razón que nos motivó a seleccionar justamente a estas tres investigaciones, en lugar de cualesquiera otras, es que aún cuando existen coincidencias en sus objetivos, la información que proporcionaron fue empleada para la invención de técnicas de corrección ortográfica muy diferentes entre sí, cada una de las cuales describiremos a continuación.

En el año de 1983, Yannakoudakis y Fawthrop buscaron una caracterización general de los errores ortográficos, esto es, buscaron reglas específicas que las faltas de ortografía tienden a seguir, para luego preparar un sistema de corrección ortográfica basado en esas reglas. Con tal idea en mente, comenzaron por construir primero una base de datos de 1'377 errores ortográficos, recopilados de muy variadas fuentes, entre las que se cuentan un bloque de texto escrito en la Universidad de Brown, y utilizado para un análisis computacional del inglés americano, hecho por Kucera y Francis en 1967, así como otro bloque de texto de 60'000 palabras escritas por tres miembros de la Universidad de Bradford, quienes, por cierto, se consideraron a sí mismos como dueños de una pésima ortografía. Al final del trabajo, Yannakoudakis y Fawthrop concluyeron que una porción considerable de los errores ortográficos del idioma inglés, puede ser corregida por la simple utilización de un conjunto de diecisiete reglas heurísticas propuestas por ellos, doce de las cuales están relacionadas con el incorrecto uso de las consonantes y las vocales en los grafemas (con motivo de su estudio, definieron un grafema como una sucesión de letras correspondientes a un fonema, por ejemplo, la palabra inglesa *that*, incluye los grafemas *th*, *a* y *t*), mientras que las otras cinco restantes son relativas a la producción de sucesiones erróneas de letras en los vocablos. Sus descubrimientos sobre la mala utilización de las consonantes incluyen afirmaciones tales como que la *h* es, frecuentemente, omitida de las palabras que contienen los grafemas *ch*, *gh*, *ph* y *rh*, como sucede en el caso de escribir *tecniques* por *techniques*. También hallaron que es muy común eliminar una consonante cuando esta debe de escribirse doble, por ejemplo, al teclear *hiting* por *hitting*. Las reglas características de las faltas de ortografía relacionadas con la producción equivocada de sucesiones de letras, incluyen notas por nosotros ya conocidas, como que los errores mecanográficos se deben, en su mayoría, al hecho de teclear letras adyacentes o cercanas en el teclado, en lugar de las letras correctas, o bien, por golpear dos teclas al mismo tiempo.

También en el año de 1983, Pollock y Zamora se abocaron a la tarea de descubrir tendencias probabilísticas que indicaran, entre otras cosas, cuáles letras y qué posiciones de las palabras están mayormente involucradas con errores ortográficos. Para desarrollar su investigación, Pollock y Zamora utilizaron máquinas lectoras de bases de datos de los servicios de recuperación de información bibliográfica, así como 50'000 palabras erradas extraídas de un total de veinticinco millones de palabras de texto escolar y científico, proveniente de siete bases de datos del *Chemical Abstracts Service*. Los principales descubrimientos obtenidos podríamos enumerarlos como sigue:

(1) Solamente el 0.2% de las palabras del texto analizado, resultó ser ortográficamente erróneo.

(2) El 94% del total de errores ortográficos fueron sencillos.

(3) El 34% del total de errores ortográficos fueron supresiones.

(4) El 23% del total de errores ortográficos se presenta en la tercera posición de una palabra.

(5) Con excepción de unos cuantos errores ortográficos poco frecuentes, la mayoría de las faltas de ortografía difícilmente se repiten en un mismo texto.

El quinto y último inciso de estos descubrimientos, nos permite reflexionar sobre el hecho de que en la

mayoría de las aplicaciones de los sistemas de corrección ortográfica, los correctores no son usados para rectificar los escritos de personas cuya ortografía pudiera calificarse de pésima, y ni tan siquiera mala. En multitud de situaciones, estos correctores sólo servirán para limpiar a los textos de errores tipográficos, y demás fallas inconscientes, que cualquier persona comete al escribir. Por lo tanto, quizá sean esos los errores sobre los que vale la pena dirigir nuestra atención de forma contundente, si es que queremos un buen sistema de corrección ortográfica.

Para finalizar, en 1990, M. D. Kernighan revisó cuarenta y cuatro millones de palabras de texto noticioso de la agencia *Associated Press*. Usó el programa *Spell* de *UNIX* para localizar palabras mal escritas en el texto, y creó una conveniente técnica para hallar todas las posibles palabras del diccionario formadas por una, y sólo una, inserción, supresión, sustitución o transposición de las letras en las palabras mal escritas. Por supuesto, encontró palabras erradas que a través de este método jamás se convirtieron en palabras del diccionario, porque contenían más de un error sencillo, también halló palabras mal escritas que se convirtieron en más de una palabra bien escrita, por ejemplo, *acress*, la cual se transformó en *actress*, *acress* y *access*. Pero lo más importante de este trabajo, es que aparecieron veinticinco mil errores ortográficos para los cuales sólo existía exactamente una posible corrección. A continuación, Kernighan procedió a preparar una lista de veinticinco mil parejas de palabras, la cual incluía, en cada par, uno de los errores ortográficos de corrección exclusiva, y su correspondiente corrección. Esta lista le sirvió para abastecer de información a una serie de tablas de frecuencia de errores ortográficos y matrices de confusión, para cada una de las cuatro clases de errores ortográficos simples: inserciones, supresiones, sustituciones y transposiciones. Entre otras cosas, Kernighan halló que la *a* fue incorrectamente sustituida por la *e* en 238 ocasiones, que la *s* fue incorrectamente agregada después de la *e* en 436 casos, que la *t* fue incorrectamente suprimida después de la *i* en 231 veces, y que la cadena *it* fue incorrectamente tecleada en vez de la cadena *ti* en 48 ocasiones. Todas estas frecuencias fueron usadas para estimar la probabilidad de ocurrencia de cada error potencial.

Con esta sección, concluimos este segundo capítulo, donde hemos descubierto que con ayuda de investigaciones profundas sobre los patrones de semejanza en la comisión de errores ortográficos, los estudiosos han llegado a producir sistemas de corrección ortográfica con porcentajes de error peculiarmente escasos, como el 0.05% asignado a las noticias editadas por cierta oficina de prensa, y otros no muy afortunados, como el 38% propio de algunas aplicaciones de gran implicación fonética. El siguiente paso a dar en este momento, consistirá en abundar en el conocimiento de las metodologías que les permiten a los programadores, aprovechar todos estos descubrimientos en apoyo del diseño de sistemas de corrección ortográfica independiente.

CAPÍTULO III
TÉCNICAS DE CORRECCIÓN INDEPENDIENTE

Sea cual sea la técnica de corrección ortográfica independiente que se utilice, sus programadores tendrán, necesariamente, que trabajar en las siguientes tres tareas: detectar errores, generar candidatos a servir de corrección y clasificar a dichos candidatos. Normalmente, las técnicas de corrección ortográfica resuelven, o cuando menos intentan resolver, cada una de estas tres tareas en procesos separados, los cuales se ejecutan en serie dentro de un módulo central. Por lógica, el primer proceso que siempre se lleva a cabo, está encargado de la simple detección de errores, y suele estar basado en la búsqueda de las palabras del texto en diccionarios de palabras válidas, o bien, en la verificación de todos los n-gramas correspondientes a las palabras del texto. Por su parte, el proceso de generación de correcciones potenciales, utiliza usualmente un diccionario, o en su defecto, una nutrida base de datos de n-gramas propios del idioma, con el objeto de localizar uno o más términos susceptibles de ser considerados como corrección. Finalmente, el proceso de clasificación de correcciones, suele involucrar alguna técnica especializada en la comparación de cadenas. Este tipo de técnicas, son conocidas con el nombre de **medidas de similitud léxica**, y facilitan la comparación entre los vocablos que representan a los errores ortográficos, y las palabras que representan a sus posibles correcciones. En algunos otros casos, el proceso de clasificación se lleva a cabo a través de estimaciones probabilísticas, por ejemplo, a través de cálculos sobre la verosimilitud de las correcciones potenciales. También conviene mencionar los casos en los que se acostumbra omitir a este tercer proceso, situación que se presenta, sobre todo, en aplicaciones de procesamiento de texto, con la intención de dejar que sea el propio usuario, quien decida cuál de todos los candidatos debe de ser seleccionado en calidad de corrección. Otro tipo de metodologías, en particular aquellas de reciente creación, vinculan los datos probabilísticos con los procedimientos de redes neuronales, pretendiendo combinar a los tres procesos en uno solo. Típicamente, estas otras metodologías implementan una medida de similitud, o una adecuada función de probabilidad, que les permita comparar cualquier cadena del texto con cualquier palabra del diccionario, o por lo menos con una partición de él. Luego, la realización completa de la rutina de comparaciones, genera, para cada cadena del texto, una lista de términos del diccionario, cuya diferencia con la cadena leída es la menor posible. Si la palabra que ocupa la primera posición de la lista no es idéntica a la cadena de entrada, entonces, se asume la existencia de un error potencial, y de inmediato se procede a suponer que la lista constituye a sus posibles correcciones.

De la misma manera que en el caso de las técnicas de exclusiva detección de errores ortográficos, los n-gramas de frecuencia estadística acostumbran utilizarse, básicamente, para aplicaciones de reconocimiento óptico de caracteres, aunque cabe observar que aún en ese tipo de aplicaciones, no es común que los n-gramas se usen sin

el apoyo de otros métodos alternativos. La razón es que se ha demostrado que la práctica aislada del análisis de los n-gramas, no es suficiente para corregir ningún texto, de modo que, en la mayoría de los casos, el análisis de n-gramas se complementa con el desarrollo de enfoques ascendentes (*"bottom-up"*) y descendentes (*"top-down"*), que proporcionan una conveniente interacción entre el empleo de los diccionarios (aspecto ascendente de la metodología), y los n-gramas de frecuencia estadística (aspecto descendente). No obstante, no muchos sistemas de corrección ortográfica realizan su trabajo de esta forma, porque cronológicamente hablando, al mismo tiempo que se perfeccionó el análisis de los n-gramas, también se practicaron investigaciones que culminaron con la creación de una creciente variedad de nuevos métodos, entre los que podríamos destacar a los siguientes: métodos de edición de mínima distancia, métodos basados en la similitud de claves inteligentemente asociadas a las palabras, métodos derivados de las reglas de ortografía, y toda una serie de diferentes aplicaciones cuyo fundamento es provisto por la teoría de probabilidades. Lo que haremos a partir de este capítulo será, justamente, estudiar a todos estos métodos, pues ellos representan a las técnicas de corrección ortográfica independiente que mayores éxitos han acumulado. Una manera apropiada de organizar este material en una redacción didáctica, dividida en secciones que faciliten el aprendizaje de toda esta información, sería la siguiente:

(1) técnicas de mínima distancia de edición;
(2) técnicas de claves similares;
(3) técnicas basadas en reglas;
(4) técnicas basadas en n-gramas;
(5) técnicas probabilísticas;
(6) técnicas de redes neuronales.

Debe advertirse que no todas las técnicas existentes en la actualidad, pueden ser ubicadas en algún lugar único en esta lista, del mismo modo que vale la pena aclarar que la mayoría de las metodologías para las que no abrimos ningún espacio particular, son el resultado de la mezcla de unas y otras técnicas, y que más bien han sido desarrolladas para aplicaciones muy especiales, y hasta cierto punto restringidas. De cualquier forma, cada una de las seis técnicas antes enumeradas, serán descritas durante las próximas páginas, en el orden marcado por los incisos que las acompañan, distribuyendo las explicaciones a lo largo de los tres capítulos siguientes, para que el material resulte lo mejor organizado posible. Esperamos ser lo suficientemente legibles, con la intención de que podamos conocer todas las metodologías, entender sus sistemas de generación de candidatos a servir de corrección, y adquirir información sobre sus distintos procesos de clasificación de candidatos.

§1. Técnicas de edición de mínima distancia

El concepto de **distancia de edición mínima**, puede ser bien entendido con la definición dada por R. A. Wagner y M. J. Fischer en 1974, en su artículo *The string-to-string correction problem*. Según Wagner y Fischer, la distancia de edición mínima es el mínimo número de operaciones de edición requeridas para transformar una cadena en otra, entendiendo por operaciones de edición a los actos de inserción, supresión, transposición y substitución de letras. El primer algoritmo de corrección ortográfica hecho a través de una técnica de edición de mínima distancia, fue implementado por F. J. Damerau en 1964, y está basado en el cálculo de una distancia de edición mínima entre una cadena ortográficamente inaceptable, y las palabras de un diccionario. Veamos ahora un poco del desarrollo histórico de esta metodología.

Más o menos en la misma época en que Damerau desarrolló su sistema, de hecho sólo un par de años después, V. I. Levenshtein diseñó un algoritmo similar para la corrección de supresiones, inserciones y transposiciones simples. Diez años más tarde, Wagner introdujo, por primera vez, la aplicación de nociones de programación dinámica a los sistemas de corrección ortográfica por mínima distancia de edición, con el objeto de

incrementar la eficiencia computacional de sus anteriores desarrollos. Su trabajo resultó por demás exitoso, y culminó con la publicación de un conocido artículo en la materia, titulado *Order-n correction for regular languages*. A raíz de este trabajo, los algoritmos de programación dinámica empezaron a ser estudiados a profundidad, para otro tipo de aplicaciones de los patrones de semejanza, como son por ejemplo, la comparación de sucesiones macromoleculares (digamos *RNA* y *DNA*), la comparación de señales audibles en los mecanismos de reconocimiento de voz, y la comparación cromática de gases para utilidad del análisis espectral. Tal vez una de las aplicaciones más interesantes de este tema, fue analizada por A. V. Aho en 1990, en un artículo sobre la eficacia de los algoritmos basados en patrones de semejanza, en el que señaló que la herramienta de comparación de archivos del sistema *UNIX*, llamada *diff*, funciona por medio de la combinación de aspectos de programación dinámica y de la técnica de la mínima distancia de edición. Todas estas aplicaciones y algunas otras más, aparecen descritas en una colección de artículos editados por D. Sankoff y J. B. Kruskal en 1983, bajo el nombre de *Time warps, string edits, and macromolecules: The theory and practice of sequence comparision*.

Sin abandonar la investigación relativa a la programación dinámica, y con una mayor experiencia en el asunto, Wagner generalizó, en 1974, con la ayuda de Fischer, el algoritmo de Levenshtein, de modo que pudiera también llevar a cabo la corrección ortográfica de errores de calidad múltiple. Algunas otras variantes algorítmicas debidas a R. Lowrance y R. Wagner en sus trabajos de 1975, permitieron perfeccionar el sistema de R. A. Wagner con modificaciones adicionales, mismas que favorecieron, entre otras cosas, el intercambio de letras no adyacentes con fines de corrección. Nuevos y mejores agregados de perfeccionamiento fueron desarrollados por C. K. Wong y A. K. Chandra en 1976, por T. Okuda, E. Tanaka y T. Kasai, también en 1976, y por R. L. Kashyap y B. J. Oomen en 1981. La forma en que todas estas técnicas calculan la distancia mínima de edición, es, normalmente, a través de una medida conocida como la **métrica de Damerau-Levenshtein**, en honor de los dos grandes precursores de esta rama de la investigación ortográfica. En el año de 1982, M. J. Hawley, con la intención de diseñar un corrector ortográfico para la interfase del lenguaje de comandos de *UNIX*, estudió esta métrica, revisando las diferentes implementaciones de los investigadores antes mencionados, y su trabajo es reconocido como una de las mejores producciones derivadas de este tema.

En la mayoría de los casos, los algoritmos que usan el método de la mínima distancia de edición, requieren m comparaciones entre la cadena erróneamente escrita y el diccionario de palabras válidas, donde m es el número de vocablos del diccionario. Sin embargo, algunas optimizaciones del método han conseguido reducir ese número de comparaciones. En este sentido, M. Mor y A. S. Fraenkel obtuvieron, en 1982, un muy eficaz tiempo de respuesta en su sistema de corrección ortográfica de distancia de edición mínima. La primera etapa del trabajo de este par de investigadores, consistió en determinar que en la aplicación para la cual ellos trabajarían, las supresiones eran el tipo más común de error sencillo, y luego diseñaron su sistema con ayuda de este interesante descubrimiento. La estrategia de Mor y Fraenkel fue la siguiente: guardar cada palabra p del diccionario un total de $[x]+1$ veces, con $[x]$ igual a la longitud de p. En el diccionario, una vez aparecería cada palabra correctamente escrita, mientras que el resto de las veces, aparecería la palabra con alguna de sus letras omitida, por supuesto con el afán de simular un posible error por supresión. Finalmente, durante el proceso de corrección, el sistema evaluaba, para cada error detectado, una función de dispersión que conducía hacia entradas del diccionario en donde podían hallarse, en lugar de una palabra correcta, los posibles errores derivados de la supresión de alguna letra en una palabra válida. Esta sistema constituye una interesante prueba del trabajo académico al servicio de las aplicaciones particulares. Obsérvese que este método, sólo garantiza su funcionalidad para dominios del lenguaje que satisfacen las condiciones que Mor y Fraenkel, tomaron como base para el desarrollo de su sistema. Más aún, lo que dentro del ambiente ortográfico en general, podría haber significado un paso hacia atrás, para esta aplicación particular representó un avance. En fin, como sucede con la mayoría de los descubrimientos computacionales, así como con las técnicas de programación, el método de la distancia mínima de edición apenas fue inventado, comenzó a servir de fuente para una multitud de nuevos métodos y técnicas híbridas, las cuales resultan de la mezcla de diferentes intentos de solución para un mismo problema. Quizá la más utilizada de todas las metodologías surgidas de la técnica de la distancia mínima de edición, es la que podríamos llamar técnica "inversa", y que fue programada, por primera vez, en 1971, por R. E. Gorin, en su corrector ortográfico para la computadora *DEC-10*. En 1983 I. Durham, D. A. Lamb y J. B. Saxe también hicieron uso de la técnica "inversa" para su corrector de lenguaje de comandos. Posteriormente, M. D. Kernighan en 1990, y K. W. Church y W. A. Gale en 1991, utilizaron la técnica "inversa"

para generar candidatos a corrección en sus sistemas de basados en teoría de probabilidades. La técnica "inversa" funciona así: una vez que una palabra es identificada como ortográficamente errónea, se asume que posee un error ortográfico sencillo. Entonces es procesada por diferentes subrutinas, una por cada tipo de error ortográfico sencillo. Por ejemplo, primero se piensa en la posibilidad de que el usuario haya cometido un error por supresión. Con esta idea en mente, en cada posición de la palabra se le inserta una letra diferente, y cada una de las nuevas palabras generadas es buscada en el diccionario. Si alguna de ellas es hallada en la lista de palabras válidas, de inmediato es aceptada como posible candidato a servir de corrección. Después se piensa en la posibilidad de haber hallado un error por inserción, y entonces se suprimen, una por una, todas las letras de la palabra, y, por supuesto, las cadenas resultantes son buscadas en el diccionario, con la intención de identificar posibles correcciones. Un procedimiento análogo se lleva a cabo con los supuestos de que se trata de un error por sustitución y de uno por transposición. En resumidas cuentas, estamos hablando de que dada una palabra errada de n letras, y un abecedario de veintiséis caracteres, la técnica "inversa" genera $26(n+1)$ palabras a través de la inserción de letras, n por la supresión de letras, $25n$ como resultado de la sustitución de letras y $n-1$ por transposiciones, para un total de $53n+25$ palabras. Luego entonces, por cada error en el texto, este método entra al diccionario $53n+25$ veces, en busca de muchos vocablos que, en su mayoría, son carentes de significado y de lugar en el diccionario, pues fueron creados de una manera puramente algorítmica, y nunca con apoyo en una conciencia léxica. Además, esta metodología sólo busca errores sencillos, y no contempla la posibilidad de errores múltiples, ante los cuales todo este trabajo resultaría inútil. Visto de esta forma, la técnica "inversa" resulta ser altamente ineficiente, pero la verdad es que muchos investigadores la han usado, y sus conclusiones no han sido tan malas, así que sus ventajas tiene. En muchos casos, la conveniencia de este método se debe a que no se usa en un sentido estricto, sino que sólo se emplea, por ejemplo, en el supuesto caso de que los errores son sólo de un tipo especial, de manera que el número de candidatos generados se reduce considerablemente.

Los *tries* también han sido usados como una alternativa a las técnicas de programación dinámica para mejorar el tiempo de búsqueda en los algoritmos de distancia de edición mínima. F. E. Muth y A. L. Tharp inventaron, en 1977, una heurística en la cual un diccionario de ortografía correcta, es almacenado, caracter por caracter, en un árbol pseudo-binario. La búsqueda examina un pequeño subconjunto de la base de datos, formado por ciertas ramas seleccionadas del árbol, y asume que los errores encontrados son de tipo sencillo, es decir, que se tratan de simples inserciones, supresiones, sustituciones y transposiciones. Muth y Tharp reportaron un altísimo porcentaje de efectividad, 97%, para su algoritmo basado en tries, cuando pretendieron corregir un texto de 1'487 palabras. Lamentablemente, ellos no especificaron el tamaño de su diccionario, ni la longitud aproximada de los vocablos que constituían su texto. En 1981 M. R. Dunlavey describió, en su artículo *On spelling correction and beyond*, un algoritmo programado por él, llamado *SPROOF*, el cual guarda un diccionario como un *trie*, esto es, *"como una inmensa máquina de estado finito que reconoce a todas las palabras del diccionario, pero solamente a ellas"*. La búsqueda de correcciones procede visitando los nodos del *trie* en una creciente distancia de edición, hasta que los candidatos a corrección sean obtenidos en las hojas. Una técnica similar fue empleada por R. H. Boivie en 1981, en su programa *DA* de asistencia al directorio, mismo que sirvió de base para el corrector ortográfico *grope*. de W. D. Taylor. Vale la pena mencionar que este sistema *grope*, fue evaluado en 1990 por K. Kukich, en una aplicación de síntesis de texto a voz, y encontró que éste es capaz de corregir, eficazmente, un 25% de 170 faltas de ortografía que involucran errores múltiples, y un 63% del total de faltas de ortografía en palabras de menos de cinco caracteres. Además, observó que con un diccionario de 1'142 palabras, es posible corregir el 62% de todos los errores ortográficos.

La distancia de edición mínima ha sido aplicada, prácticamente, a todas las tareas de la corrección ortográfica, a saber, edición de texto, interfases de lenguajes de comandos, interfases de lenguaje natural, etcétera. La precisión con la que la distancia mínima de edición corrige, varía de acuerdo con las aplicaciones y las implementaciones. Damerau reportó que para un total de 964 errores ortográficos sencillos, todos ellos en palabras de cinco o más caracteres, y un diccionario de 1'593 palabras correctas, la técnica de la distancia corrige el 95% de los errores. Su porcentaje de corrección global, esto es, cuando se enfrenta ante una corrección de texto real, en el cual puede haber errores ortográficos de calidad múltiple, fue del 84%. Por su parte, Durham, Lamb y Saxe, en su corrector para un lenguaje de comandos, obtuvieron un porcentaje global de eficiencia de tan sólo el 27%, pero diseñaron un algoritmo simple y rápido de corrección de errores sencillos, que no hostiga al usuario con

intentos equivocados de corrección, ni con detecciones de posibles errores controlados. Uno de los detalles más curiosos de este sistema de Durham, Lamb y Saxe, es el hecho de que usó un diccionario de apenas cien palabras, aunque claro, como hemos dicho, no fue programado para corregir un texto de lenguaje común, sino para un lenguaje de comandos. Reiteramos que con todo y su miserable porcentaje del 27%, Durham y los otros autores, reportaron un alto grado de satisfacción en sus usuarios, debido, precisamente, a que su sistema no produjo exagerados conjuntos de candidatos a servir de corrección, ni tampoco detecciones falsas, con la cual se demuestra que ningún porcentaje no lo es todo.

§2. Técnicas de claves similares

La idea en la que se fundamenta la técnica de claves similares, consiste, simplemente, en asignarle a todas las cadenas una clave, de tal manera que aquellas palabras que sean muy parecidas entre sí, tengan la misma clave, o cuando menos una muy semejante. Luego entonces, si a una palabra erróneamente escrita le asignamos una cierta clave, esta nos proporcionará un apuntador hacia todas las palabras correctas y parecidas a ella, que ocupan un lugar en el diccionario, y que le servirán de candidatos a corrección. Una de las mayores ventajas de esta metodología, es que su velocidad de ejecución es muy elevada, pues no tiene que comparar directamente a todas las faltas de ortografía con todas las palabras del diccionario. Esta es una de las razones por las que muchos programadores la han utilizado. Entre sus principales aplicaciones se encuentra el sistema *SOUNDEX*, patentado por M. K. Odell y R. C. Russell en 1978, para utilidad de un corrector ortográfico fonético. Este sistema le asigna a cada palabra una clave formada por su primera letra, y una sucesión de dígitos determinada por el resto de sus letras en correspondencia con la siguiente tabla:

Letras	Dígito Correspondiente
A, E, I, O, U, H, W, Y	0
B, F, P, V	1
C, G, J, K, Q, S, X, Z	2
D, T	3
L	4
M, N	5
R	6

Los ceros obtenidos son eliminados, y los dígitos repetidos que quedan juntos también se eliminan, para que aparezcan una única vez. Por ejemplo, la palabra *kangaroo* (canguro en inglés) pasará por las siguientes transformaciones hasta obtener su clave

$$kangaroo \rightarrow k0520600 \rightarrow k526$$

Por su parte, la palabra *kangaru*, posible error ortográfico cometido al tratar de escribir kangaroo siguiendo la intuición auditiva, tomará la clave

kangaru → k052060 → k526

De este modo, la cadena *kangaru* será señalada como falta de ortografía, y su corrección aparecerá en la misma dirección que su clave nos proporcionará. Análogamente, la clave del error ortográfico *Busch* (*Busch→B0220→B22→B2*) será la misma que la de su corrección *Bush* (*Bush→B020→B2*). En el español podría suceder que algunas faltas de ortografía, tales como *allé*, no tuvieran la misma dirección que sus correcciones, en este caso *hallé*, pues la clave del error empezaría con *a*, mientras que la de la corrección con *h*. De hecho, tanto en el español como en el inglés, las claves de un error y de sus posibles correcciones, difieren mucho cuando la primera letra del error es distinta de la primera letra de la corrección. Para obtener una mayor eficacia de esta técnica en el español, podríamos modificar las reglas, haciendo, por ejemplo, que la primera letra de la palabra forme parte de su clave siempre y cuando no sea la letra *h*, y en el caso de que la primera letra sea una *h*, que esta se elimine y que su clave se inicie con su segunda letra. También convendría hacer que las vocales acentuadas tengan asignado el mismo dígito que las vocales no acentuadas, en este caso el cero. Esto último debería llevarse a cabo toda vez que la vocal acentuada apareciera después de la primera letra, porque si se encuentra al principio convendría transformarla en la misma vocal pero sin acentuar. Por supuesto que la *u* con diéresis sería tratada de manera similar que las vocales acentuadas. Otra manera inmediata de perfeccionar la técnica, sin importar el idioma para el cual se trabaje, sería, por algún método, reducir los posibles candidatos a corrección, para no hacer que simplemente aparezcan todas las palabras guardadas en una misma dirección, aunque en el peor de los casos esto último no sería del todo despreciable, baste citar que el sistema *SOUNDEX* funcionaba justamente así.

Pollock y Zamora usaron sus descubrimientos del estudio de 1984, aquellos sobre los 50'000 errores ortográficos de siete bases de datos del *Chemical Abstracts Service*, para idear una técnica de claves similares llamada *SPEEDCOP*, para la corrección automática de errores ortográficos sencillos. Su metodología comienza igual que la de Odell y Russell en el *SOUNDEX*, pero después de obtener la clave de una cadena, se reunía a todas las palabras con claves anteriores y posteriores cercanas en un diccionario ordenado con respecto a las claves. Luego, a todas esas palabras se les aplicaba una transformación propia de un error sencillo, esto es, una inserción, una supresión, una sustitución o una transposición, y, finalmente, se les revisaba, una por una, hasta dar con la primera que coincidiera con la cadena errónea capturada. Este método es claramente más lento que el que consiste en proporcionar toda una lista de palabras con la misma clave, o con claves parecidas, pero es evidentemente más preciso, y más adecuado para los sistemas de corrección automática.

El funcionamiento del sistema *SPEEDCOP*, se verifica asignándole dos diferentes claves a cada palabra: una clave maestra y una que podríamos llamar clave de omisión de candidatos a corrección, y que sólo se utiliza para seleccionar al más apropiado de los candidatos a corrección. La asignación de ambas claves fue cuidadosamente diseñada, a través de descubrimientos estadísticos, con el objeto de alcanzar una máxima capacidad de corrección para la aplicación específica que trabajaron, la cual incluía, por cierto, muy pocos errores de causa fonética. Tanto la clave maestra como la clave de omisión estaban formadas por la aparición, una sola vez, de todas las letras de la palabra acomodadas con respecto a un orden especial. Para la clave maestra el orden fue el siguiente: la primera letra de la cadena ocupa también el primer lugar de la clave, después le siguen únicamente las consonantes en orden de aparición, cuidando que ninguna letra aparezca más de una vez en la clave. Finalmente, todas las vocales de la palabra, también por orden de aparición y sin repetirse. Para la clave de omisión, las consonantes se colocan primero, pero en orden inverso de omisión, estos es, dejando primero a aquella de las consonantes que, según Pollock y Zamora, es omitida con menor frecuencia en el idioma. De acuerdo con los hallazgos de estos dos investigadores, las consonantes del idioma inglés son omitidas en el orden revelado por la siguiente lista:

R S T N L C H D P G M F B Y W V Z X Q K

Al final de la clave de omisión se colocan las vocales en orden de aparición en la cadena original, pensando en que ellos habían descubierto que los errores ortográficos tienden a preservar el orden de aparición de las vocales. Como ejemplo de asignación de la clave de omisión, podríamos decir que a la palabra *chemical* le corresponde la clave *MHCLEIA*. Obsérvese que ésta misma clave es asignada a los errores *chemicial* y *chemcial*, mientras que la clave para el también error *chemcal* es muy parecida, a saber *MHCLEA*.

El algoritmo del sistema *SPEEDCOP* procede conforme a cuatro pasos básicos, cada uno de los cuales es aplicado sólo si el paso previo falla, al momento de producir la corrección requerida para el error hallado. Los cuatro pasos son:

(1) Consultar un diccionario de errores ortográficos comunes. Recordemos que el sistema de Pollock y Zamora, es uno de los muy pocos correctores ortográficos que incorpora una rutina que revisa una lista de palabras comúnmente mal escritas.

(2) Obtención de la clave maestra, según las reglas ya enunciadas.

(3) Obtención de la clave de omisión, de acuerdo con los descubrimientos estadísticos antes citados.

(4) Aplicación de una rutina que busca posibles concatenaciones entre palabras erradas.

El origen del paso cuatro, último del algoritmo, fue discutido en una de las situaciones mencionadas al principio de este capítulo, cuando dijimos que la palabra *ala* puede ser erróneamente escrita al pretender escribir la frase *a la*. Por supuesto, en este caso, la concatenación de las palabras *a* y *la* generó otra palabra del diccionario, pero esto no tiene por qué ocurrir siempre. Algunas concatenaciones no están incluidas en la lista de palabras válidas. Pollock y Zamora mencionaron que el último paso incrementa el porcentaje de corrección apenas de un 1% a un 2%. Sin embargo, lo más importante del cuarto paso es la impresionante precisión que alcanza cuando se utiliza después de los otros tres.

Al final del trabajo de análisis del corrector *SPEEDCOP*, Pollock y Zamora concluyeron que su técnica tiene una precisión de entre el 77% y el 96%, sobre los errores ortográficos sencillos, lo cual es muy apropiado si consideramos que, en promedio, el 94% de los errores ortográficos son sencillos, al menos en las aplicaciones de los estudios de estos dos investigadores. En total, según Pollock y Zamora, el *SPEEDCOP* corrige entre el 74% y el 88% de todos los errores ortográficos, independientemente de que estos sean sencillos o múltiples. El *SPEEDCOP* es un sistema bastante completo para su aplicación específica, situación que se refleja incluso en el tamaño de su diccionario: una lista de 40'000 palabras en total.

Otra ingeniosa técnica de corrección ortográfica basada en claves similares, fue inventada en 1972 por P. Tenczar y W. Golden. Ellos trabajaron para el *Laboratorio de investigaciones de educación basada en computadoras* de la Universidad de Illinois, y su proyecto consistía en la preparación de un sistema de tutoría computacional llamado *PLATO*. De la misma manera que el *SPEEDCOP*, Tenczar y Golden construyeron una clave especial para cada palabra, y, desde un principio, el diccionario estuvo ordenado con respecto a las claves. Sin embargo, a diferencia de otros sistemas, las claves del *PLATO* fueron hechas pensando en toda una serie de rasgos cognoscitivos humanos observados por sus autores. Por tal motivo, las claves asignadas a las palabras, valoraban grandemente los siguientes aspectos:

(1) la longitud de la palabra,
(2) la letra inicial de la palabra,
(3) las letras contenidas en la palabra,
(4) el orden de las letras contenidas en la palabra, y
(5) la pronunciación silábica de la palabra.

Cada uno de estos cinco incisos fue representado por un campo de bits en la clave de similitud, la cual, entre otras cosas, puede ser perfeccionada a placer, pues según la opinión de Tenczar y Golden, si se desea pensar en detalles adicionales que se agreguen a la lista de los cinco incisos ya existentes, éstos bien podrían ser considerados y representados con facilidad en la clave, agregando simplemente más campos de bits. Por otra parte, cuando *PLATO* inicia su ejecución, primero busca en su diccionario la palabra tecleada por el usuario, trabajo que puede hacer aplicando un procedimiento que calcule la clave de esa palabra, para luego utilizar un segundo

procedimiento que ejecute una búsqueda parecida a la binaria, pero en este caso no a lo largo de un diccionario ordenado alfabéticamente, sino a través de un diccionario ordenado por claves. Muchas veces, de hecho siempre que la palabra tecleada está mal escrita, es imposible hallar su respectiva clave en el diccionario, pero, ante tal contrariedad, *PLATO* está implementado de manera que su proceso de búsqueda, sirve también para proporcionar la clave más similar a la requerida, en el caso de que no sea posible conseguir exactamente a la deseada. Desde luego, esa clave, considerada como la más similar a la correspondiente al vocablo recibido por el sistema, será después el principal candidato a corrección de *PLATO*.

Para finalizar su investigación, Tenczar y Golden sometieron su sistema a un examen de eficiencia, con la intención de medir sus alcances. Para tal efecto, hicieron que *PLATO* realizara su trabajo de corrección sobre un archivo formado por un diccionario de errores ortográficos comunes, y hallaron que su técnica es capaz de corregir, sin falla alguna, un porcentaje aproximado del 95% de los errores pertenecientes a cualquier muestra tomada de aquel diccionario. Como segunda parte de su examen, Tenczar y Golden hicieron que su sistema corrigiera las quinientas palabras más usadas en el idioma inglés, pero escritas con mala ortografía. El resultado fue satisfactorio, pues en el peor de los casos, cuando el sistema hizo correcciones deficientes, la diferencia entre la corrección adecuada y la hecha por el sistema, fue, en promedio, de apenas una letra.

Nadie puede negar que el trabajo de Tenczar y Golden, así como el de los equipos de investigación que dieron origen al *SOUNDEX* y al *SPEEDCOP*, han sido muy exitosos en el ambiente científico, tanto por la calidad de sus investigaciones como por las aportaciones que sus estudios han hecho a la materia, y ni que decir de la probada excelencia que sus sistemas han obtenido. Pero la verdad es que hasta la fecha ningún sistema ha terminado de convencer a todos los estudiosos de este tema, y no es así tan sólo porque todos sean muy exigentes, sino más bien porque siempre queda el deseo de mejorar lo poco o lo mucho que se ha conseguido. Esta es, justamente, una de las causas que a la larga motivan, en los diversos terrenos de la investigación académica, la combinación de diferentes metodologías de trabajo, que para nuestro caso se traducen en diferentes técnicas de corrección ortográfica, precisamente con la finalidad de aprovechar las ventajas proporcionadas por unas y por otras, de la misma forma que con la ilusión de evitar también las desventajas de unas y las imperfecciones de otras. Es así como se originó el algoritmo conocido con el nombre de **reconstrucción atómica** (*token reconstruction*), el cual vincula la técnica de las claves similares con la técnica de la mínima distancia de edición, utilizando una medida patentada por A. K. Bocast en 1991. La vinculación entre las dos técnicas de corrección ortográfica, consiste en el hecho de que la forma de medir la distancia entre dos cadenas, no esté basada en las operaciones de edición necesarias para transformar a la primera en la segunda, sino en la similitud existente entre las dos.

Dada una cadena ortográficamente errónea y una palabra extraída del diccionario, el algoritmo de la reconstrucción atómica comienza por *medir la similitud* entre una y otra. Dicha medida es obtenida a partir de un promedio tomado de la suma de cuatro factores valuados empíricamente, y que determinan la presencia de subcadenas de longitud máxima, compartidas por el error ortográfico y por la palabra del diccionario. La reconstrucción atómica se apoya también en la estrategia de particionar el diccionario, con la intención de mejorar su tiempo de ejecución. En la versión original de este algoritmo, se dividió al diccionario en paquetes de palabras que poseen el mismo caracter inicial c, y la misma longitud n. De esta forma, cuando llegaba el momento de localizar a los candidatos a servir de corrección para un determinado error, la búsqueda de estos principiaba en el paquete c_i, n_j, y se extendía hasta el paquete c_i^*, $n_j \pm 1$, es decir, se iniciaba buscando entre las palabras empezadas con la letra c_i que tenían una longitud n_j, y concluía con las palabras iniciadas con la letra c_i^* que tenían un caracter de más o de menos que las primeras. Esta tarea se lleva a cabo, inicialmente, con las palabras que cuentan con pequeñas diferencias de longitud, y luego se prosigue con aquellas que poseen una primera letra distinta. El éxito de la reconstrucción atómica se remite a los porcentajes de efectividad que proporciona: el 78% de los candidatos a corrección que genera en calidad de primera opción, corresponden exactamente a la corrección deseada. Cuando menos esta fue la calificación obtenida por el corrector debido a Bocast, una vez que examinó un total de 123 faltas de ortografía, entre las que se contaban quince errores ortográficos múltiples. Esta calificación es en verdad exitosa, pues representa una exactitud superior a la conquistada por cuatro de los más populares correctores ortográficos comerciales.

Aún sin disponer de mucha experiencia en este tema, consideramos que la reconstrucción atómica es el mayor logro alcanzado en el terreno de las técnicas que podríamos llamar híbridas, en el sentido de que mezclan una metodología con otra. La verdad es que, al menos por lo que se refiere a las claves similares y a las distancias de edición, no se ha conseguido mayor éxito que este. Por esta razón, no quisimos concluir esta pequeña sección sin mencionarla, aunque bien podríamos abrir una nueva sección titulada técnicas híbridas.

§3. Técnicas basadas en reglas

Las técnicas basadas en reglas sirven para producir algoritmos, o mejor dicho, métodos heurísticos, que combinan eficientemente el conocimiento adquirido en las investigaciones referentes a los patrones de semejanza en la comisión de errores ortográficos, con una inteligente programación de rutinas que verifican el cumplimiento de las reglas de ortografía del idioma. El ambicioso objetivo de esta metodología, consiste en hacer que la computadora se convierta en una maestra de educación elemental, capaz de corregir las faltas de ortografía de sus alumnos, no precisamente con el apoyo irremediable de un diccionario, pero sí con el apropiado dominio de las reglas de ortografía. El proceso de generación de candidatos a corrección que posee esta técnica, se basa en la aplicación sucesiva de todas las posibles reglas de corrección ortográfica propias del lenguaje, sobre cada error ortográfico detectado, con la intención de guardar aparte cada una de las palabras válidas que resulten de semejante actividad. Cuando alguna de las reglas de ortografía, digamos la regla r, consigue hacer que una palabra errónea q se convierta en una nueva palabra p, que sí pertenece al diccionario, entonces, p es nombrada candidato a corrección, y, al mismo tiempo que se integra en una lista, se le asigna un puntaje numérico, basado en estimaciones predefinidas, en la mayoría de los casos, sobre la probabilidad de que el usuario haya cometido exactamente el error particular que la regla r corrigió. Posteriormente, todas aquellas palabras que como p hayan sido obtenidas, se someten a una ordenación, desde luego con la idea de determinar cuál de todas ellas es la que finalmente será tomada como corrección.

En el año de 1983, Yannakoudakis y Fawthrop diseñaron un programa de corrección ortográfica de este tipo. Usaron las deducciones de su antes comentado estudio de 1'377 errores ortográficos, y después formaron un conjunto de reglas de ortografía que los podía corregir a todos. En el momento en que concluyeron su sistema, lo usaron para corregir un bloque de 1'534 errores ortográficos, y contaron con el auxilio de un diccionario de 93'769 palabras. Debido a que algunas de sus reglas estaban relacionadas con el conocimiento de la posible longitud de la palabra correcta, el corrector utilizó un diccionario particionado en subconjuntos, tomando como parámetro para hacer la partición, la longitud de la palabra y la inicial de la misma. Para su proceso de generación de candidatos, este sistema buscaba en el diccionario palabras que diferían del correspondiente error ortográfico por solamente uno, o a lo mucho dos, errores de ortografía, y que a la vez cumplían con todas las reglas de corrección implementadas. Cabe observar que la búsqueda de estas palabras no se realizaba a lo largo de todo el diccionario, sino nada más en una sola de sus particiones, la cual era adecuadamente escogida. En el supuesto caso de que fueran hallados numerosos candidatos a servir de corrección, estos se ordenaban por la probabilidad de ocurrencia de las reglas bajo las cuales el error capturado se convertía en el candidato escogido.

Yannakoudakis y Fawthrop hallaron que la corrección apropiada para el error detectado, estaba en la partición seleccionada para buscarla en un 75% de las veces, lo cual significó, en términos de los 1'534 errores ortográficos que se trataron de corregir, que en 1'153 ocasiones se halló la corrección requerida en el lugar en el que se le buscó. Ahora bien, de esas 1'153 ocasiones, la palabra correcta fue devuelta como primera opción de corrección, en el 90% de los casos. Así pues, el promedio de exactitud total de este sistema de Yannakoudakis y Fawthrop, fue del orden del 68% (90% del 75%).

Otro sistema de corrección ortográfica basado en reglas, indispensable de mencionar por sus objetivos y su ambiente de desarrollo, fue el creado por L. G. Means en el año de 1988. Si algunos correctores ortográficos mencionados por nosotros hasta este momento, han sido considerados ambiciosos en sus pretensiones, este sistema

hecho por Means no se queda atrás, al contrario, los supera y por mucho. Means diseñó un corrector ortográfico para un sistema de entrada de datos en lenguaje natural, el cual contaba con una amplia incidencia de abreviaturas y expresiones en jerga especializada. Este sistema primero revisaba la presencia de algunas violaciones comunes en el lenguaje inglés, tales como, por ejemplo, no duplicar adecuadamente la consonante final de la palabra antes de agregar la terminación *ing* en la construcción de los gerundios. Este tipo de errores podían ser fácilmente detectados a través de la inclusión de algunas reglas morfológicas conocidas. Como siguiente paso, el corrector consultaba un conjunto de reglas de expansión de abreviaturas, porque siempre que el sistema leía una abreviatura del texto la consideraba como una palabra cualquiera, pero al no hallarla en el diccionario, la tachaba como error ortográfico. Por tal motivo, era necesario un procedimiento que se encargara de averiguar si es que las abreviaturas podían ser expandidas hasta generar verdaderas palabras del diccionario. En caso de fallar también en este segundo paso, el sistema procedía a aplicar todas las posibles transformaciones relativas a errores ortográficos sencillos sobre la cadena considerada como falta de ortografía, inclusive pensando en la posibilidad de que el usuario hubiese omitido teclear un espacio en blanco en el lugar adecuado, o bien, el caso en que hubiese tecleado un espacio de más, situación que se conoce con el nombre de error de separación de palabras (*split-word error*).

Hoy en día, todavía estamos muy lejos de sentirnos satisfechos con los correctores ya existentes, los cuales, entre otras cosas, no son capaces de realizar corrección dependiente de contexto, de corregir justo el error que nos echa a perder el texto, y ni siquiera de evitarnos tantos rechazos de palabras que para nosotros están bien, o de realizar la selección apropiada de la palabra exacta que representa la corrección a nuestro error. Así que a estas alturas, pensar en la posibilidad de crear un sistema de corrección que pueda incluso ser capaz de revisar abreviaturas, o de analizar expresiones escritas en caló, para después determinar si es que son válidas, es un proyecto muy exagerado. No obstante, es conveniente mencionarlo, a sabiendas de que los objetivos del trabajo de Means, siguen siendo motivos de investigación, como él mismo lo afirma en su *Proceedings of the 2nd applied natural language processing conference*.

CAPÍTULO IV
TÉCNICAS MATEMÁTICAS

Lo que más nos asombra cuando comparamos la matemática de nuestro tiempo con la de épocas anteriores es la extraordinaria diversidad y el aspecto imprevisto de los caminos y los atajos por los que esta ciencia se ha embarcado, es el desorden aparente con que ejecuta sus marchas, son las maniobras y los continuos cambios de frente.
(Pierre Boutroux)

Para un matemático con conocimientos elementales de análisis matricial, hablar de los n-gramas es tanto como caminar sobre terreno conocido, sin nada que temer y con el control en las manos. En un sentido estricto, un arreglo n-grama no es otra cosa sino una cualquier matriz, como todas aquellas que siempre son descritas en los teoremas del álgebra lineal, y que si acaso sirven para descubrir que en el idioma existen sucesiones de letras inaceptables, pues no es más que una coincidente irrupción alfabética en el campo de las ciencias exactas. Ahora bien, la verdad es que las matemáticas comienzan a hacerse presentes, o digamos inevitablemente presentes, en el tema de las técnicas de corrección ortográfica, desde el momento mismo en que se mencionó por primera vez la idea de medir semejanzas entre las palabras, esencia de las metodologías de la mínima distancia de edición. A este respecto, podemos garantizar que sin matemáticas, dicha metodología nunca se habría sido inventada, dado que la noción de evaluación de distancias nunca se le habría ocurrido a un redactor, o a un corrector de estilo, por mucha que fuese su experiencia en la revisión de textos. Sin embargo, para un matemático, el cálculo de distancias, las normas y los espacios métricos, son algo tan elemental que flota en su ambiente, y basta con alzar la mano para alcanzarlo. Quizá algún día puedan decirse cosas como que la corrección de la falta p es la palabra q, porque el *Teorema de la representación léxica de Krhäsnërberg*, o *Rhukehenköff*, o algún otro impronunciable, establece que en todo espacio léxico, cuya dimensión sea igual a un número primo, la función *corrección* es una transformación funcional homeomórfica, que lleva a una combinación lineal en el núcleo en otra combinación lineal en la imagen, determinada por el valor de la transformación en los elementos de una base del núcleo, y, en consecuencia, q corresponde a p bajo la *corrección*. Entonces, seguramente, un marciano del año 5'000 se preguntará si la automatización ortográfica, fue un tópico computacional surgido en apoyo de la siempre imperfecta humanidad, o bien, una proeza topológica. En fin, si la metodología de la distancia mínima de edición, requería de las matemáticas para satisfacer con números a sus preguntas aritméticas, los n-gramas y los métodos probabilísticos, como se observará más tarde, necesitan de las matemáticas para mucho más que eso, pues descansan sobre la base de modelos que a los académicos les ha tomado años enteros teorizar.

§1. Técnicas basadas en n-gramas

Recientemente, los bigramas, los trigramas y hasta los monogramas, han sido utilizados en una multitud de sistemas de reconocimiento de texto y de exclusiva corrección ortográfica. Nada más por citar algunos ejemplos de la utilización de n-gramas, vale la pena mencionar sus siguientes aplicaciones:

(1) Captura de la sintaxis léxica de un diccionario, en correctores adaptados para dispositivos de reconocimiento óptico de caracteres.

(2) Sugerencia de correcciones válidas en correctores especializados en el mismo tipo de dispositivos citados en el inciso (1).

(3) Aportación de información significativa sobre las características léxicas de las palabras del idioma, situación que favorece la invención de métricas de similitud apropiadas, así como la creación de convenientes métodos de obtención de claves de acceso, o funciones de dispersión para los diccionarios propios de cada implementación.

(4) Representación de palabras y errores ortográficos como vectores de características léxicas, para los cuales nuevas y tradicionales medidas de distancia vectorial pueden ser útiles, con el propósito de localizar, u ordenar, candidatos a servir de corrección ortográfica.

Una clara explicación sobre el uso tradicional de n-gramas en los procesos de corrección de las lecturas de los dispositivos de reconocimiento óptico, fue proporcionada por E. M. Riseman y A. R. Hanson en su trabajo publicado en 1974, bajo el título de *"A contextual postprocessing system for error correction using binary n-grams"*. Primeramente, trataremos de seguir este texto como base para la redacción de nuestro capítulo, y, conforme avancemos, lo iremos complementando con algunas otras observaciones obtenidas de diferentes fuentes. Un buen comienzo, sería iniciar con la descripción de las técnicas propias de aquellos sistemas que pretenden emplear a los n-gramas como única y exclusiva fuente de información para sus procesos de corrección.

En resumen, podemos decir que Riseman y Hanson explicaron que cuando se pretende utilizar n-gramas, primero se debe dividir al diccionario en subdiccionarios, que contengan siempre palabras de una misma longitud, y luego construir arreglos n-gramas binarios posicionales para cada uno de los subdiccionarios. Se acostumbra decir que este tipo de arreglos capturan **la sintaxis de un diccionario**, porque son capaces de responder a la pregunta *"¿existe alguna palabra en el diccionario que posea las letras α y β en las posiciones i y j, respectivamente?* Para saber si cualquier cadena leída por el lector óptico es ortográficamente correcta, basta con verificar que todos sus n-gramas tengan valor uno. Cuando las cadenas cuentan con un único error sencillo, entonces, tendrán al menos un valor de cero en alguno de sus arreglos n-gramas binarios. Debe observarse que si simplemente buscamos las palabras del texto en el diccionario, podremos saber si es que dichas palabras son válidas o no, pero nunca podremos saber cuál es la posición exacta del error que las invalida, y por eso es que para poder producir su corrección, es indispensable aplicarle todas las posibles operaciones de edición para errores sencillos. Sin embargo, el método de los n-gramas sí puede indicarnos la posición exacta del error en una palabra errada, lo que nos concede una cierta ventaja adicional cuando pretendamos corregirla. Más aún, con un poco de inteligencia y habilidad, es incluso posible encontrar la corrección deseada con ayuda de los n-gramas, al menos en el caso de errores de sustitución. En efecto, si una palabra tiene más de un valor de cero en su lista de n-gramas, la posición del error es indicada por el índice matricial que posee el valor de cero en todos los n-gramas. Los candidatos a corrección pueden ser ubicados por la intersección de los renglones o las columnas especificadas por el índice compartido por los n-gramas incorrectos. Si la intersección genera un único n-grama con valor uno, entonces el error puede ser corregido inmediatamente, pero si más de una corrección potencial puede ser hallada, la palabra se rechaza en calidad de ambigua.

Al igual que la mayoría de los investigadores, Riseman y Hanson examinaron su sistema después de

programarlo. Utilizaron para el examen un conjunto de 2'755 palabras, de seis letras cada una, y que contenían errores ortográficos. Sus resultados indicaron que con ayuda de arreglos trigramas posicionales, fueron capaces de detectar el 98.6% de los errores existentes, de los cuales pudieron corregir el 62.4%. Como se puede observar, aunque no obtuvieron un muy elevado puntaje en términos de corrección, sí alcanzaron un lugar envidiable en el campo de la detección. Por otra parte, debemos hacer notar que todos los sistemas disponen, en el nivel de corrección, de porcentajes menores a los de detección, y no muy superiores al 50%. Así que dentro de lo que cabe, un 62.4% de corrección para el sistema de Riseman y Hanson, hace que lo podamos clasificar de exitoso.

Como fin de su investigación, Riseman y Hanson apuntaron que una desventaja menor de la técnica de los n-gramas, tal y como ellos la manejaron, es que su forma de generar correcciones puede llegar a producir palabras inexistentes, puesto que no las genera a través del diccionario, sino con la simple inspección de sucesiones válidas de caracteres. Otra desventaja de este método, a la cual sí podríamos considerar de suma importancia, es que es una técnica diseñada especialmente para resolver el problema de los errores de sustitución, que son los más frecuentes en el reconocimiento óptico de caracteres, pero que ni siquiera se puede predecir que tan bien vaya a funcionar ante otro tipo de faltas, tales como inserciones, supresiones, transposiciones y errores de separación de palabras. Cuando menos la investigación de Riseman y Hanson no condujo a ninguna posible observación al respecto.

En otro orden de ideas, dos importantes técnicas, basadas no solamente en el desarrollo de sistemas, sino también en la utilización de implementos de hardware, han sido propuestas para explotar, en paralelo, las bases de datos proporcionadas por los n-gramas. La primera de ellas fue realizada en 1977, por J. R. Ullmann, quien diseñó una metodología para hallar, dada una falta de ortografía, todas las palabras válidas que difieren de ella por a lo más dos inserciones, supresiones, sustituciones e inversiones, a través del procesamiento en paralelo de n-gramas binarios. La segunda técnica de este tipo fue elaborada en 1987, por J. Henseler, J. C. Scholtes y C. R. J. Verdoest, y se basó en la implementación de un algoritmo paralelo de reconocimiento óptico de caracteres y de corrección ortográfica, para una máquina *16-processor hypercube*. Este desarrollo usó una base de datos alimentada por trigramas de frecuencia estadística, en oposición a los trigramas binarios. Un examen de esta metodología sobre una pequeña muestra de texto, manifestó resultados fabulosos: 95% de exactitud en el reconocimiento inicial de caracteres, y 100% de precisión del reconocimiento después de la corrección.

Para el año de 1983, R. C. Angell, G. E. Freund y P. Willett publicaron un escrito especial para su discusión durante un seminario, al cual nombraron *Automatic spelling correction using a trigram similarity measure*. En este trabajo, explicaron el diseño de una técnica de corrección ortográfica que calcula una medida de similitud, basada en el número de trigramas binarios no posicionales comunes a una cadena ortográficamente errada p, y a una palabra del diccionario p'. En términos de ecuaciones, la medida de similitud fue dada por la función

$$S(p, p') = \frac{2c}{n+n'},$$

donde c es el número de trigramas comunes a p y p', y n y n' son las longitudes de p y p', respectivamente. Angell, Freund y Willett, se refieren a esta función S como "el bien conocido **coeficiente de Dice**". El sistema que crearon también poseía un diccionario en el que las palabras contaban con un índice inverso, y utilizaba a los trigramas como claves de acceso a él. El funcionamiento del sistema puede resumirse del siguiente modo: primero, se calculan todos los trigramas de la cadena ortográficamente errada, para localizar a todas aquellas palabras del diccionario que cuentan con al menos un trigrama en común con ella. La función de similitud léxica es luego aplicada, solamente sobre ese subconjunto del diccionario.

La técnica de Angell, Freund y Willett, obtuvo un puntaje de exactitud del 76%, sobre un total de 1'544 errores ortográficos, utilizando un diccionario de 64'636 palabras. Posteriormente, los tres investigadores estuvieron particularmente interesados en analizar la capacidad que este corrector tenía para trabajar en contra de todos los

diferentes tipos de errores. Sus resultados fueron los siguientes: el corrector trabaja muy bien ante errores de omisión e inserción, adecuadamente con errores de sustitución, y muy pobremente con errores de transposición. Aunque la técnica no se derrumbó ante el problema de la longitud de las palabras, sus autores aceptaron que no se puede esperar que tenga un excelente funcionamiento con vocablos de escasa longitud, porque en ellos, un error sencillo puede dejar intactos algunos trigramas inválidos. El promedio de longitud de las palabras que analizaron en su examen, fue de 8.4 caracteres.

Otro detalle que limita el buen funcionamiento del sistema de Angell, Freund y Willett, es que cuando la cadena que representa a la falta de ortografía, está totalmente contenida en una palabra válida, o viceversa, el *coeficiente de Dice* tiende a producir resultados ilógicos. Para resolver esta contrariedad, los estudiosos advirtieron que la tendencia mostrada por los errores ortográficos, para diferir en la longitud de su correspondiente corrección, puede ser utilizada cambiando a la función de similitud S por la función

$$S'(p, p') = \frac{c}{\max(n, n')}.$$

El equipo halló que esta función de similitud léxica, corrige el problema de la contención de errores, sin afectar el porcentaje de corrección para errores ortográficos de calidad múltiple.

Métodos similares de corrección en base a trigramas, han sido ideados por T. Kohonen en 1980 y por T. DeHeer en 1982, para aplicaciones de edición de texto y devolución de información, respectivamente. Una técnica relacionada, llamada análisis trifono (*triphone analysis*) fue desarrollada por B. Van Berkel y K. DeSmedt, en 1988, específicamente para una interfase de lenguaje natural, la cual involucraba el continuo empleo de nombres propios, mismos que en la mayoría de los casos se convertían en faltas de ortografía, toda vez que pretendían ser escritos con ayuda de la intuición fonética. El sistema funcionaba con la aplicación inicial, sobre cada cadena errada, de un conjunto de reglas de conversión de grafemas a fonemas, para luego hacer uso de un *índice trifono inverso*, que permitía la adquisición de candidatos a corrección. Los resultados de un experimento en el que usaron a este corrector ortográfico, fueron altamente exitosos: el sistema corrigió el 94% de las palabras contenidas en un archivo formado con faltas de ortografía, derivadas del mal deletreo de 123 apellidos alemanes. Su diccionario estuvo constituido por una base de datos de 254, apellidos aleatoriamente tomados de un directorio telefónico alemán. Van Berkel y DeSmedt hicieron algo no muy usual en este tipo de investigaciones: compararon su técnica fonética en contra de otras bien conocidas técnicas de corrección ortográfica no fonética. Gracias a esto, descubrieron que su sistema superó a todos los demás, al menos cuando fueron usados para corregir el mismo conjunto de texto que mencionamos antes. El corrector que mejor calificación obtuvo ante el sistema de Van Berkel y DeSmedt, fue superado por este último por cuatro puntos porcentuales, mientras que el peor de ellos fue superado por 40 puntos.

Durante los últimos años, una gran variedad de medidas de distancia vectorial, algunas nuevas y otras más tradicionales, han sido empleadas en esta materia, por medio de la representación de palabras en forma de vectores de n-gramas. La mayoría de estas técnicas reúnen en un único procedimiento, las tareas de detección de errores, generación de candidatos y clasificación de los mismos. Las ideas básicas en cuanto al funcionamiento de todas estas metodologías, podrían ser indicadas por los siguientes dos incisos:

(1) ubicar cada palabra del diccionario en algún punto de un espacio n-dimensional de características léxicas, y luego,

(2) proyectar una cadena ortográficamente errada sobre ese mismo espacio, y medir la proximidad entre ella y sus más cercanos vecinos, los cuales, obviamente, serán sus candidatos a corrección.

Los monogramas, los bigramas y los trigramas constituyen tres posibles formas de caracterizar al espacio léxico, y las palabras, de la misma forma que los errores ortográficos, pueden ser representados como vectores en

tales espacios. Las distancias tradicionalmente utilizadas en los espacios vectoriales, los productos interiores y las distancias apoyadas en el cálculo de cosenos, proporcionan tres posibles métricas para el espacio léxico generado.

Uno de los más interesantes estudios comparados que se han hecho sobre el tema de la corrección ortográfica computarizada, es el de Kukich de 1992, realizado con el objeto de verificar cuál de los diferentes métodos de corrección, poseía una mayor exactitud durante las diferentes etapas del desarrollo de su ejecución. Al respecto de las tres métricas propuestas al final del párrafo anterior, podemos comentar que Kukich evaluó en su estudio la eficacia de las tres, disponiendo que todas corrigieran exactamente un mismo conjunto de errores, y que lo hicieran apoyadas sobre la base de un mismo diccionario, para que ninguna tuviese ventajas especiales. Conviene observar que el archivo que estos tres sistemas tuvieron que corregir, así como el diccionario que se requirió que usaran, fue, justamente, el mismo que le sirvió a Kukich para evaluar la capacidad del corrector ortográfico *grope*, basado en una técnica de edición de mínima distancia, anteriormente citado en este mismo capítulo, y que según asentamos, obtuvo un porcentaje de efectividad del orden del 62%. En cuanto al archivo que tuvieron que corregir, se debe indicar que contaba con 170 errores ortográficos, entre sencillos y múltiples, de los cuales, aproximadamente, dos tercios se hallaban repartidos en palabras de escasa longitud, digamos menos de cinco caracteres. En cuanto al léxico, su única particularidad radicó en el hecho de poseer tan sólo 1'142 palabras. Los sistemas utilizaron vectores de 790 entradas, algunas de ellas proporcionadas por monogramas y el resto por bigramas, y a través de ellos caracterizaron a las palabras y a las faltas de ortografía. Con ayuda de todo ese material, el trabajo de Kukich produjo los siguientes resultados:

(1) El sistema diseñado con la métrica del producto interior, alcanzó el más bajo de los porcentajes de exactitud, a saber, 54%. Esto significa que con semejante fuente de distancias vectoriales, se consigue corregir eficientemente, apenas un poco más de la mitad de los errores existentes en el texto.

(2) La métrica tradicionalmente utilizada en espacios vectoriales comunes, en las principales aplicaciones del álgebra y del cálculo (la métrica usual), generó, en su correspondiente sistema, un porcentaje de eficiencia del 68%.

(3) La métrica de los cosenos, obtuvo el mayor de los porcentajes de exactitud en la corrección de texto, con ayuda del mencionado diccionario. Su porcentaje fue del 75%.

Vale la pena observar que dos de estos tres sistemas de corrección ortográfica independiente, cuentan con una mayor eficiencia que el corrector *grope*, de donde podríamos atrevernos a pensar que las técnicas basadas en n-gramas tienen, de algún modo, una mejor calidad que aquellos basados en la mínima distancia de edición. Sin embargo, un juicio de esta naturaleza no puede ser dicho con tanta arbitrariedad, porque después de todo el *grope* no es más que un herramienta más de corrección, quizá la mejor y más representativa de su género, y, por supuesto, un instrumento indispensable de software para ciertas aplicaciones, pero a través del cual no se puede determinar el comportamiento de toda una técnica. Además, aún en la peor de las comparaciones, hay que reconocer que si bien la métrica de los cosenos obtuvo una mejor calificación que el *grope*, la diferencia entre uno y otro es de tan sólo trece puntos porcentuales, lo cual no puede ser suficiente para tachar de inadecuada a una técnica y de sobresaliente a la otra.

Una metodología de corrección ortográfica con apoyo de espacios vectoriales, que no se puede considerar como basada en n-gramas, pero sí altamente relacionada con ellos, es la conocida técnica de la **memoria de matrices de correlación** (*correlation matrix memory*, o bien, para abreviar, *CMM*). En esta metodología, se pretende representar a un diccionario de m palabras, por medio de una matriz de nxm, a través de vectores n-dimensionales de características léxicas. El proceso de corrección ortográfica realizado bajo un sistema de esta naturaleza, se puede resumir de la siguiente manera: una primera rutina de detección establece la existencia de posibles errores ortográficos en el texto, los cuales una vez que han sido capturados, se representan con ayuda de

un vector *n*-dimensional de características léxicas. Posteriormente, cada uno de estos vectores es multiplicado por la matriz de *nxm*, la cual caracteriza a la totalidad del diccionario. El resultado de dicho producto es, de acuerdo con las reglas del álgebra matricial, un vector *m*-dimensional, en el que, por convención, la *i*-ésima entrada corresponde a la *i*-ésima palabra del diccionario. Además, cada entrada del vector en cuestión posee un número diferente, por medio del que se indica la *correlación* existente entre cada una de las palabras del diccionario, y la falta de ortografía en proceso, esto es, el grado de semejanza entre cada una de las palabras del diccionario y la falta de ortografía detectada. De este modo, si la *j*-ésima entrada del vector dispone del valor más alto, entonces se asume que la *j*-ésima palabra del diccionario es la que más se parece a esa falta de ortografía, y, por lo tanto, es justamente ella quien se convierte en el principal candidato a servir de corrección.

Durante el año de 1992, V. Cherkassky, N. Vassilas, G. L. Brodt y H. Wechsler, hicieron detallados estudios para comparar el uso de los bigramas contra el uso de los trigramas, en el trabajo con vectores de características léxicas para modelos *CMM*. Para tal efecto, prepararon un par de archivos que contuvieran errores ortográficos sencillos generados aleatoriamente. El primero de estos dos archivos estuvo formado por palabras de longitud media, digamos que de cinco a siete letras, y el segundo fue hecho exclusivamente con palabras largas, digamos de diez a doce caracteres. Para poder llevar a cabo los procesos de corrección ortográfica, utilizaron diversos diccionarios de tamaños variables entre las 500 y las 11'000 palabras. Cherkassky, Vassilas, Brodt y Wechsler, observaron que el empleo de trigramas, proporciona excelentes porcentajes de eficiencia en la corrección basada en modelos *CMM*, a saber, porcentajes superiores al 90%. Por su parte, los bigramas producen porcentajes que van de favorables a buenos, cuando son utilizados para corregir palabras de longitud media. En efecto, los bigramas produjeron, en el peor de los casos, porcentajes del orden del 60%, mientras que en el mejor de los casos, alcanzaron porcentajes cercanos al 85% de efectividad, observándose que la variación de estos resultados, está en completa dependencia del tamaño del diccionario que fue utilizado para construir los bigramas.

Otro estudio que continúa en el mismo orden de ideas que este último, fue el también realizado por Cherkassky en el año de 1990, aunque en esa ocasión sólo fue acompañado por P. Dahl. En dicho estudio, ellos compararon el uso de monogramas, bigramas y trigramas para corrección de palabras de todo tipo de longitudes, y entre las que se encontraban presentes, errores pertenecientes a los cuatro tipos de errores sencillos. Sus resultados probaron que los bigramas y los trigramas, alcanzan los mejores porcentajes de corrección para cualesquiera de los cuatro tipos de errores sencillos, con excepción de los errores de transposición, mismos que se corrigen mejor con los monogramas, sin importar la longitud de la palabra. Precisamente, sobre los errores de transposición, cabe mencionar una magnífica investigación hecha en 1980, por G. Deloche y F. Debili, publicada bajo el largo título de *Order information redundancy of verbal codes in French and English: Neurolinguistic implications*. De acuerdo con este trabajo, tanto en el francés como en el inglés, el 96% de los anagramas producidos con palabras extraídas de diccionarios de más de 20'000 vocablos, sin contar conjugaciones de verbos ni demás inflexiones gramaticales o declinaciones morfológicas, tienen soluciones únicas. Recordemos que un anagrama es una palabra que resulta de la transposición de las letras de otra; por ejemplo, *amor* es un anagrama de la palabra *Roma*. De este modo, lo que expresa el estudio de Deloche y Debili, es que si tomamos cada una de las diferentes palabras de un diccionario de más de 20'000 vocablos, y generamos todos sus posibles anagramas, hallaremos que en el 96% de los casos, sólo la palabra original tiene sentido dentro del idioma, mientras que ninguno de sus anagramas forma parte del diccionario, y, por consiguiente, todos ellos constituyen potenciales faltas de ortografía.

Desde que la utilización de espacios de características léxicas en la programación de correctores ortográficos, comenzó su auge, muchos han sido los intentos de transformar a dichos espacios en caracterizaciones manejables, normalmente involucradas con el empleo de conocidas técnicas matemáticas. En un cierto sentido, es como si se hubiera pactado que la guerra en contra de la mala ortografía, dejará de librarse en el terreno lingüístico, para que pudiera llevarse a cabo sobre el campo de las matemáticas, en donde se tiene la esperanza de ganar la batalla final. Una de las técnicas en las que esta guerra parece ver sus más cruentas escenas, es la que se basa en el cálculo de la **matriz inversa generalizada**, mayormente conocida con el nombre de técnica *GI*. Esta metodología se basa en la aplicación del cálculo de los mínimos cuadrados, sobre una muestra de datos proporcionados por la inversa de la matriz que representa al conjunto de las palabras válidas del idioma, y cuyo objetivo es reducir, al mínimo, la dificultad de determinar cuál es la palabra que representa a la corrección más apropiada para un cierto

error. Si se desea una mayor explicación al respecto de esta metodología, se pueden consultar un par de trabajos publicados por Cherkassky, y que llevan por nombre *Fault-tolerant database retrieval using distributed associative memories*, el primero, y *Linear algebra approach to neural associative memories and noise performance of neural classifiers*, el segundo. Por supuesto que ambos artículos podrían proporcionarnos una excelente visión acerca de los intentos de conducir al tema de la corrección ortográfica hacia las profundidades matemáticas, pero la verdad es que, en el caso de la técnica *GI*, los beneficios reales obtenidos para el perfeccionamiento de los correctores, fueron prácticamente nulos. El mismo Cherkassky, junto con Vassilas y K. Fassett, demostró, en el año de 1991, que una matriz inversa generalizada se satura cuando el número de entradas del diccionario se aproxima a la dimensión del espacio léxico, situación que provoca un significativo declive en la exactitud de cualquier corrector programado bajo esta metodología. Así pues, Cherkassky, Vassilas y Fassett, concluyeron que había que dejar a un lado a la técnica *GI*, y dedicarse mejor a la programación de modelos *CMM*, los cuales resultaron ser más simples, efectivos y eficientes para las aplicaciones ortográficas.

Un esfuerzo más por interpretar matemáticamente a los espacios léxicos, lo constituye la **técnica de la descomposición en valores propios**, conocida como técnica **SVD**, por sus siglas en inglés (*Singular Value Decomposition*). Sin lugar a dudas, la descomposición en valores propios, resultó ser bastante más exitosa que la de la matriz inversa generalizada. Tal vez la investigación más conocida sobre esta técnica, es la que fue hecha por K. Kukich en 1990. De acuerdo con Kukich, la técnica *SVD*, puede ser utilizada para descomponer a una matriz que represente a todo el léxico de un idioma, en un producto de tres matrices: la primera de las tres matrices acumularía información sobre los n-gramas válidos en el idioma, y los mantendría almacenados como un vector de factores; la segunda matriz sería una diagonal, cuyos elementos estarían dados, justamente, por los valores propios de la matriz original del léxico; por su parte, la tercera matriz representaría a cada una de las palabras del diccionario como un vector de factores. Según una observación de Kukich, los factores menos importantes, y que bajo ciertas condiciones podrían resultar más un estorbo que una necesidad, pueden ser descartados, de manera que nuestro trabajo estuviera reducido al empleo de tres matrices de rango reducido, las cuales capturarían solamente las relaciones esenciales entre las palabras del diccionario. En 1990, S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer y R. Harshman hicieron uso de la técnica *SVD*, en experimentos de devolución de información, y reportaron muy favorables resultados. En aquellos experimentos, la computadora dispuso de una inmensa base de datos repleta de información bibliográfica, la cual contenía miles de documentos y palabras del diccionario, todas ellas representadas por matrices llamadas **término-por-documento**. La razón de semejante denominación tiene que ver con el hecho de que los experimentos realizados por Deerwester, y por los demás investigadores recientemente mencionados, condujeron a la elaboración de un sistema al que se le ha nombrado **indexación semántica latente** (*Latent Semantic Indexing*), debido a que la técnica *SVD* es usada para manifestar las relaciones semánticas inherentes entre los términos y los documentos.

En términos generales, podemos decir que los sistemas de corrección ortográfica que utilizan matrices para la ejecución de su trabajo, funcionan bajo la siguiente metodología:

(1) Con el apoyo de los n-gramas se construye una matriz de corrección ortográfica, y en ella las palabras son representadas por vectores monogramas y bigramas.

(2) A la matriz de corrección se le descompone en tres matrices cuyo producto es igual a ella. En la primera se representan a los n-gramas como un vector de factores, la segunda es una matriz diagonal que contiene a los valores propios de la matriz de corrección ortográfica, y la tercera posee información sobre las palabras del diccionario.

(3) Cualquier error ortográfico detectado es corregido de la siguiente manera: Primero se suman los vectores correspondientes a los n-gramas propios del error ortográfico. Después se multiplica el vector suma por la matriz de valores propios. El vector resultante determina la posición que el error ortográfico tiene dentro del espacio léxico *n*-dimensional. Una métrica cualquiera, como por ejemplo el producto interior o la distancia basada en cosenos se usa entonces, para conocer las distancias existentes entre el vector característico del error ortográfico y las palabras del

diccionario, representadas en la tercera matriz. Este último procedimiento localiza y clasifica las palabras correctas más próximas al error detectado, para que sean justamente ellas sus posibles correcciones.

Los estudios que Kukich hizo en 1990 sobre la técnica *SVD*, indican que la descomposición de la matriz de corrección ortográfica produce mejores porcentajes de exactitud en la corrección que cuando se trabaja con la matriz sin descomponer, siempre y cuando se utilicen diccionarios pequeños. De este modo, descubrió que si se usa un diccionario de 521 palabras, una matriz no descompuesta reditúa un porcentaje de exactitud de 76%, mientras que una matriz descompuesta obtiene un porcentaje de hasta 81%. Ahora bien, si se trabaja con diccionarios de gran tamaño, la descomposición de la matriz de corrección no produce ninguna mejora significativa. Puede observarse, por lo tanto, que el estudio de Kukich es consistente con el que hizo Cherkassky, en compañía de otros investigadores, para determinar el punto de saturación de las matrices inversas generalizadas.

Para finalizar la exposición de las técnicas basadas en n-gramas, mencionaremos una técnica híbrida (como dijimos al final de la sección anterior, bien podríamos abrir una nueva sección titulada *Técnicas híbridas*), que utiliza monogramas y una heurística basada en métodos de distancia vectorial y medidas léxicas. Esta metodología fue hecha por M. A. Bickel en 1987, mismo año en que la publicó en su artículo *Automatic correcction to misspelled names: A fourth-generation language approach*. Bickel diseñó su técnica híbrida para una aplicación computacional administrativa, y de intercambio de información, en la cual se manejaron fuertemente los nombres propios de los empleados adscritos a una cierta corporación, razón por la cual fue indispensable corregir la ortografía de todos los nombres propios antes de procesarlos. Así pues, el diccionario consistió de aproximadamente mil nombres propios, y los mismos nombres sirvieron como claves de acceso al diccionario. Bickel representó cada uno de los nombres válidos como un vector monograma. El valor de la *i*-ésima entrada del vector monograma fue cero, si la letra correspondiente a la *i*-ésima entrada del vector no aparecía en el nombre propio, mientras que si la letra sí tenía al menos una aparición en el nombre, el valor de la *i*-ésima entrada del vector monograma era algún número entero entre el tres y el nueve. La forma de conocer exactamente el número que le correspondía a las entradas representativas de letras sí existentes en el nombre, estaba relacionada con el hecho de que cada letra del alfabeto, tenía asignado un número entero entre el tres y el nueve, de modo que si la letra correspondiente a la *i*-ésima entrada del vector, sí aparecía en el nombre, el número que se la adjudicaba era, justamente, el número propio de esa letra. El valor entero correspondiente a cada letra del alfabeto, fue predeterminado de acuerdo con su frecuencia de aparición en el diccionario de nombres, tomando en consideración que a las letras menos frecuentes, se les iban a asignar los números más grandes, debido a que ellas eran más representativas del nombre. Cuando el sistema de Bickel recibía un nombre propio, lo convertía en un vector monograma binario, y la técnica asumía que la primera letra del nombre tecleado era correcta, razón por la cual limitaba su búsqueda, únicamente, a la porción del diccionario que pertenecía a esa letra. En seguida, el sistema calculaba una medida de similitud léxica entre el nombre insertado por el usuario, y cada uno de los nombres de una cierta porción del diccionario, a través de la obtención del producto interior de los vectores correspondientes.

A Bickel se le debe considerar como uno de los investigadores más exitosos en el ambiente de la corrección ortográfica automática, aún cuando su desarrollo técnico resultaría por demás inútil, si se pretendiera corregir con él cualquier otra cosa que no fueran los nombres propios de su base de datos. Sin embargo, a Bickel no le interesaba hacer nada más que eso, y su sistema fue creado con esa idea en mente, de modo que no se le podía exigir que sirviera como un corrector de uso generalizado. Además, a nadie le interesa disponer de un impresionante material de software que no pueda hacer el trabajo que nosotros requerimos. Antes al contrario, lo que todos quisiéramos tener es algo que nos auxiliara justamente en aquello que sí nos hace falta, aunque eso mismo no sirviera para nada más. En ese sentido, para la aplicación específica para la que Bickel trabajó, su sistema alcanzó magníficos porcentajes de eficiencia, siempre superiores al 95% de exactitud, a lo largo de varios exámenes practicados, y en donde se pretendía que la misma computadora devolviera los nombres que el usuario había querido teclear, aunque los hubiese tecleado mal.

§2. Técnicas basadas en probabilidad

Cronológicamente hablando, las técnicas basadas en n-gramas abrieron, de manera natural, el camino para el paso posterior a las técnicas basadas en cálculos probabilísticos, tanto en la materia del reconocimiento de texto como en los paradigmas de la corrección ortográfica. En trabajos publicados en las últimas décadas, sobre investigaciones hechas en esta área de estudios, queda claro que sólo dos clases de probabilidades han sido explotadas: Las **probabilidades de transición** y las **probabilidades de confusión**. Las probabilidades de transición determinan la probabilidad de que una letra dada, o una sucesión de letras, sea seguida por cualquier otra letra. Las probabilidades de transición son dependientes del lenguaje, en el sentido de que es justamente el lenguaje, quien sirve de parámetro para determinar el valor de una cierta probabilidad de esta clase. A las probabilidades de transición se les acostumbra llamar **probabilidades de Markov**, por el hecho de que se sabe que el lenguaje puede ser visto como una *fuente de Markov*. Las probabilidades de transición pueden ser estimadas a través del análisis de n-gramas de frecuencia estadística recopilados de grandes cuerpos de texto.

Las probabilidades de confusión, son estimaciones hechas para medir la frecuencia con la que una letra dada es errróneamente escrita, o sustituida, por alguna otra letra del alfabeto. Las probabilidades de confusión son más que nada dependientes de la forma en que ha sido capturado el texto, porque, tal y como ha sido explicado en los párrafos precedentes, sobre todo en los del capítulo anterior, un mecanógrafo y un reconocedor óptico, cuando se equivocan en la captura de un texto, no cometen el mismo tipo de errores por sustitución. Más aún, los mecanógrafos expertos y los inexpertos no se puede garantizar que incurran en los mismos errores, además de que en el caso de los dispositivos de reconocimiento óptico de caracteres, las diferentes clases de dispositivos utilizan diferentes técnicas para la lectura de caracteres, razón por la cual cada dispositivo en especial debe contar con su propia distribución de probabilidades de confusión. De hecho, conviene observar que, en el caso del reconocimiento óptico de caracteres, un mismo dispositivo de reconocimiento puede llegar a tener diferentes distribuciones de sus probabilidades de confusión, una por cada diferente tipo y tamaño de letra que está capacitado para leer. Precisamente, por el hecho de que cada método particular de captura de texto posee sus propias probabilidades de confusión, algunas veces a las probabilidades de confusión se les nombra **características de canal**.

Un modelo de probabilidad de confusión para un dispositivo específico, puede ser obtenido alimentando al dispositivo de una muestra conveniente de texto, y tabulando las estadísticas de sus errores. A este proceso se le acostumbra llamar fase de **entrenamiento** (*training*), porque muchas veces este trabajo es practicado para el desarrollo posteriores sistemas de corrección ortográfica. Algunos métodos alternativos de perfeccionamiento del reconocimiento óptico, proponen que el dispositivo genere un vector de veintiséis entradas, una por cada letra del alfabeto inglés, con el objeto de almacenar en él información sobre la probabilidad de que cada letra del alfabeto aparezca en el texto al momento del reconocimiento óptico. En ese caso, las estimaciones de probabilidad necesarias bien podrían ser interpretadas como probabilidades de confusión.

Las probabilidades de confusión basadas en errores generados por humanos, son simplemente llamadas **probabilidades de error**, y son, por ejemplo, las que se derivan del análisis de textos capturados por mecanógrafos. Esta clase de probabilidades han sido estimadas a partir de la extracción de estadísticas, relativas a los errores existentes en grandes muestras de texto producido por los humanos. La obtención de esta clase de probabilidades, con todo y que proporcionan una muy importante fuente de información para el desarrollo de sistemas de corrección ortográfica, ha sido poco explotada por los estudiosos. Apenas hace unos cuantos años, se le ha concedido un mayor interés al trabajo con estadísticas sobre los errores generados por humanos. Sin embargo, los investigadores del reconocimiento óptico de texto, han trabajado intensivamente con información probabilística. Esto mismo, les ha permitido comprobar que, actualmente, la utilización exclusiva de las técnicas de la ciencia de las probabilidades en los métodos de corrección es insuficiente, pero la combinación de información probabilística con técnicas de búsqueda en diccionarios, rinde significativos beneficios a la corrección ortográfica.

Toda área del estudio especializado, cuenta con sus propios pioneros de la ciencia, y en esta área, como pioneros de la investigación probabilística en las técnicas de reconocimiento de texto, no podríamos dejar de citar

a W. W. Bledsoe y a I. Browning, quienes desarrollaron, en 1959, una técnica que todavía es practicada hoy en día, en cierto tipo de aplicaciones. Su técnica se divide en dos etapas:

(1) La primera etapa corresponde a un reconocimiento individual de caracteres. Es en esta etapa cuando se aprovecha para construir, para cada caracter de una palabra capturada por el dispositivo de reconocimiento, un vector especial de probabilidad que cuenta con veintiséis entradas.

(2) La segunda etapa es utilizada para un reconocimiento de palabras completas. Es en esta etapa cuando se acostumbra usar un diccionario, con el objeto de ayudarle al sistema a seleccionar los caracteres individuales cuya combinación de probabilidades de aparición maximiza la probabilidad de producir una palabra existente en el diccionario.

Como era de esperarse, el reconocimiento de palabras completas perfecciona al reconocimiento de caracteres individuales, pues la utilización del diccionario concede una ventaja adicional en el proceso de corrección de errores cometidos durante la etapa del reconocimiento individual. Inclusive, permite despejar posibles dudas en cuanto a la verdadera naturaleza del caracter leído, sobre todo en el supuesto caso de leer textos poco legibles.

En cuanto al cálculo de probabilidades, conviene mencionar que las técnicas de Bledsoe y Browning utilizan la bien conocida regla de Bayes, para el cómputo de una probabilidad *a posteriori* para cada palabra del diccionario, a partir de una probabilidad *a priori* para caracteres individuales. Digamos que X representa una palabra del diccionario y que Y representa una cadena que el dispositivo de reconocimiento óptico devolverá como producto de su lectura. Entonces, la regla de Bayes establece que

$$P(X/Y) = \frac{P(Y/X) \cdot P(X)}{P(Y)} \ldots (1),$$

donde $P(X \mid Y)$ representa la probabilidad condicional de que X sea la palabra correctamente escrita, $P(Y \mid X)$ es la probabilidad de leer a Y cuando X es la palabra correcta, y $P(X)$ y $P(Y)$ son las probabilidades independientes de las palabras X y Y. Determinar que X es la palabra del diccionario con una mayor probabilidad de haber sido escrita, dado que la cadena Y fue recibida por el dispositivo, equivale a maximizar la función

$$G(Y/X) = \log P(Y/X) + \log P(X) \ldots (2),$$

donde $P(X)$ es frecuentemente tomada del monograma de probabilidades de la palabra X. La probabilidad *a posteriori* correspondiente a cada palabra del diccionario puede ser rápidamente calculada, a través de las estimaciones de probabilidad *a priori* de los diferentes caracteres individuales y del empleo de la ecuación

$$\log P(Y/X) = \sum_{i=1}^{i=n} \log P(Y_i/X_i) \ldots (3),$$

donde n es la longitud de la palabra del diccionario, en este caso X, y X_i y Y_i representan los caracteres individuales de las palabras X y Y, respectivamente.

Con el ánimo de exhibir un ejemplo de todo lo que hasta aquí hemos dicho, podemos declarar lo siguiente. Si suponemos que el reconocedor óptico leyó la palabra *DOQ*, y que la palabra *DOG* forma parte del diccionario, entonces,

$$G(DOQ/DOG) = logP(D/D) + logP(O/O) + logP(Q/G) + logP(DOG) \ldots (4)$$

En la metodología diseñada por Bledsoe y Browning, las estimaciones de la probabilidad de aparición de cada caracter en la cadena devuelta son suplidas por el mismo trabajo del dispositivo de reconocimiento, el monograma de frecuencia de la palabra se ignora y, simplemente, la palabra con la máxima probabilidad se selecciona.

Los requerimientos computacionales de la técnica de Bledsoe y Browning, crecen en la misma medida en que crece el diccionario, de manera que el procesamiento de diccionarios de gran tamaño es altamente inconveniente, aún cuando estos sean divididos de acuerdo con la longitud de las palabras. En el año de 1983, D. J. Burr amplió la metodología de Bledsoe y de Browning, con la intención de diseñar un sistema de corrección ortográfica para una aplicación de reconocimiento de texto escrito a mano. Dado que Burr no podía prescindir de un enorme diccionario, adicionó la técnica de Bledsoe y Browning con una rutina de procesamiento morfológico, de modo que su diccionario fuese más pequeño, pues contaba sólo con las raíces de las palabras, y dejaba fuera a todas sus inflexiones, conjugaciones verbales y declinaciones morfológicas. Más acerca del interesante trabajo de Burr puede ser consultado en su artículo *Designing a handwriting reader*.

En 1987, S. Kahan, T. Pavlidis y H. S. Baird combinaron la utilización de probabilidades de confusión con los métodos de búsqueda en diccionario, para revisar textos producidos por dispositivos de reconocimiento óptico. Su trabajo funcionaba del siguiente modo: cuando una cadena producida por el dispositivo lector era rechazada por el detector de errores ortográficos, el sistema generaba, sucesivamente, candidatos a servir de corrección, basadas en probabilidades de confusión obtenidas a través del entrenamiento (*training*) antes referido, hasta que una palabra del diccionario fuera producida, o bien, hasta que un límite de búsqueda fuese excedido. Kahan y su grupo reportó un porcentaje de exactitud del orden del 97%, en la tarea de reconocer caracteres, aunque anunciaron que no tomaron en cuenta parejas de caracteres indistinguibles por su dispositivo lector, como por ejemplo, *O* y *O* (cero y la letra *O*) y *1* y *l* (uno y la letra *l*).

Con el paso de los años, se han diseñado algunas técnicas especiales con las que se ha pretendido, entre otras cosas, utilizar únicamente a las probabilidades de transición y confusión, sin el apoyo adicional de un diccionario, para que el hecho de no tener que perder valiosos instantes durante las búsquedas en el diccionario, permita reducir el tiempo de ejecución. En 1987, A. R. Hanson, E. M. Riseman y E. Fisher reportaron algunos experimentos interesantes, en los que combinaron la probabilidad con la utilización de trigramas binarios posicionales. Sus resultados finales resultaron ser más satisfactorios de lo que se esperaba. Primeramente, descubrieron que los trigramas binarios posicionales, por sí solos, corrigen más del 50% de las faltas de ortografía que aparecían en el texto que examinaron como prueba, de manera que cuando el sistema fue mejorado con las técnicas probabilísticas, su porcentaje de eficiencia llegó a ser más que notable.

G. T. Toussaint en 1978, al igual que J. J. Hull y S. N. Srihari en 1982, publicaron estudios separados, en los que se exponen ideas generales al respecto de la posibilidad de diseñar técnicas de corrección basadas, exclusivamente, en probabilidades de confusión y de transición. Por ejemplo, argumentan la posibilidad de que la fórmula de Bayes sea usada, pero sólo a partir de la combinación de las dos clases de probabilidades en cuestión. Pero quizá el método más eficiente, y mayormente utilizado, para combinar las probabilidades de confusión y transición, es el llamado **algoritmo de Viterbi**, y que fue preparado con el auspicio de la teoría de la programación dinámica. Este algoritmo salió a la luz pública en el año de 1973, con la impresión del trabajo de G. D. Forney junior, titulado justamente *The Viterbi algorithm*. Una excelente exposición del mismo algoritmo, pero con la intención de ser aplicado en las tareas de corrección ortográfica, se halla en la antes mencionada investigación de Hull y Srihari en 1982, la cual fue publicada con el nombre de *Experiments in text recognition with binary n-gram and Viterbi algorithms*. En esencia, el algoritmo de Viterbi consiste en la representación de la estructura del léxico (la cual se obtiene a través de las probabilidades de transición), y de las características de canal del dispositivo de reconocimiento particular que será usado (las cuales pueden ser derivadas de las probabilidades de confusión), por medio de una gráfica dirigida. Un nodo inicial y un nodo terminal caracterizan, respectivamente, el inicio y el final de una cierta palabra, mientras que los nodos intermedios sirven para proporcionar estimaciones sobre la

probabilidad de aparición de letras individuales, y las aristas de la gráfica manifiestan las probabilidades de transición existentes entre las diferentes letras. Posteriormente, con ayuda del algoritmo de Viterbi, la gráfica es eficientemente recorrida hasta encontrar la sucesión de letras que suma la probabilidad más alta de ser una palabra correcta, dadas las estimaciones hechas sobre la lectura particular proporcionada por el dispositivo utilizado en esta tarea, y dadas las probabilidades de transición conocidas en el idioma.

En la actualidad, los anales de la investigación relativa a la corrección ortográfica automatizada, dispone de numerosas modificaciones hechas sobre el algoritmo de Viterbi, pero ninguna ha podido evitar de un modo apropiado y definitivo a la principal de las desventajas que tiene el algoritmo. En efecto, desde la primera vez en que este algoritmo fue usado en el ambiente de la corrección ortográfica, y hasta el día de hoy, pasando por todas sus modificaciones, siempre ha conservado el problema de ser capaz de generar palabras que no existen en el idioma. Recordemos que el algoritmo simplemente devuelve la cadena cuyas letras agrupadas en un orden apropiado suman la más alta probabilidad de corrección, pero una sucesión de letras, poseedora de la más alta de las probabilidades posible, no siempre coincide con una palabra válida, por ilógico que esto sea. Por esta razón es que estudiosos como Toussaint, Hull y Srihari han concluido que la exactitud de las técnicas basadas exclusivamente en las probabilidades de confusión y transición, deja mucho que desear, y la mayor parte de las veces es mediocre.

Otra técnica más, que también pretende explotar a las probabilidades de confusión y de transición sin el apoyo de ningún otro recurso más, es la conocida con el nombre de **técnica de relajación** (*relaxation technique*), descrita por A. Goshtasby y R. W. Ehrich en 1988, en su artículo *Contextual word recognition using probabilistic relaxation labeling*. Esta técnica revisa a cada cadena capturada por el dispositivo de reconocimiento óptico, de manera que las palabras que sean reportadas como leídas ya hayan sido ortográficamente revisadas. Su funcionamiento procede de la siguiente manera, una vez recibido el primer caracter de la palabra, el algoritmo termina de construir el resto de la palabra, cotejando los caracteres leídos con las estimaciones de probabilidad de aparición de cada caracter. Una fórmula conocida con el nombre de fórmula de relajación de Rosenfeld es utilizada para ajustar, iterativamente, las probabilidades de cada caracter como una función de las probabilidades de transición de los caracteres adyacentes. El número de caracteres candidatos a convertirse en parte de la palabra capturada por el dispositivo, puede ser limitado a través de cotas adecuadamente escogidas. La complejidad computacional que el algoritmo de la relajación manifiesta para cada palabra, es del orden de IOn^2, donde n es el número de las letras de la palabra.

La mayor de las desventajas que tiene la técnica de la relajación es que, de la misma forma que el algoritmo de Viterbi, puede llegar a producir palabras inexistentes en el idioma. Tan sólo por citar un ejemplo, vale la pena mencionar que en un examen que le fue practicado a un sistema programado bajo esta metodología, la computadora convirtió a la palabra correcta *biomass* en la palabra inexistente *biomess*, debido a que la sucesión de letras constitutiva de la primera de estas palabras es menos probable que la sucesión que genera a la cadena biomess. Goshtasby y Ehrich consideran que la técnica de la relajación podría ser mejorada, si se le agrega información adicional. Por ejemplo, si el sistema es adicionado con una base de datos formada por bigramas que aporten información sobre las sílabas válidas y más probables en el idioma, entonces, durante la ejecución del algoritmo de la relajación, aumentarían, en cada iteración, las probabilidades de los caracteres que son candidatos a ser parte de la cadena leída, y que pueden formar sílabas compatibles con el idioma. También aceptaron que el proceso de relajación puede ser perfeccionado con el acompañamiento de alguna técnica basada en la utilización de diccionario, con todo y que esto va en contra del propósito original de evitar las búsquedas en diccionarios.

Otra forma de utilizar la información probabilística en las aplicaciones de corrección ortográfica, con muy buenos resultados, por cierto, es como preprocesador de correctores. La actividad en este terreno comenzó en 1988, cuando T. Oshika y otros tres investigadores a su cargo, implementaron un procedimiento de esta naturaleza, al cual denominaron **Modelo de Markov Oculto** (*Hidden Markov Model*, o bien, *HMM*, para abreviar). Este procedimiento sirvió para preclasificar apellidos de acuerdo con información étnica conocida y estudiada anteriormente. Los procedimientos *HMM* fueron hábilmente diseñados por Oshika y su equipo, uno para cada uno de los cinco diferentes idiomas involucrados con la aplicación requerida, basándose para ello en las probabilidades de transición de sucesiones de letras tomadas de muestras de 2'000 apellidos de cada lenguaje. Durante la ejecución del sistema,

Viterbi. Esta técnica fue examinada con un texto de 5'000 palabras (25'000 caracteres en total), escritas en máquina de escribir y por otros medios, además de contar con diferentes tamaños y estilos de letra. Su porcentaje total de exactitud en el reconocimiento óptico de caracteres, excedió el 98%, y su exactitud en el reconocimiento de palabras completas, estuvo alrededor del 97%, sin contar errores de puntuación. Otras observaciones hechas sobre la base de este trabajo, fueron que el diccionario aumentado tiene un mejor desempeño en la corrección de textos no convencionales, como por ejemplo documentos literarios y técnicos, que el diccionario no aumentado. Sin embargo, en la corrección de textos convencionales, el diccionario aumentado deteriora el funcionamiento del sistema y lo conduce a porcentajes de exactitud menores al 90%.

En 1991, M. A. Jones, G. A. Story y B. W. Ballard programaron un postprocesador para el reconocimiento óptico de caracteres, el cual integró un tratamiento bayesiano de predicciones, mismas que fueron hechas a partir de una multitud de fuentes de conocimiento. En una fase de entrenamiento anterior a la utilización del sistema, Jones y los demás construyeron modelos estadísticos para el dispositivo de reconocimiento óptico de caracteres, al tiempo que también prepararon relaciones sobre el tipo de documentos que serían procesados. Sus modelos incluían información recopilada de muy variadas fuentes: las frecuencias de las letras en monogramas y bigramas, trigramas binarios y frecuencias de las letras iniciales de los diferentes n-gramas, frecuencias de los monogramas y bigramas en palabras completas y análisis sobre la puntuación y utilización de las letras mayúsculas y minúsculas.

El corrector de Jones y su equipo funcionaba en tres etapas:
(1) Generaba una lista ordenada de los posibles candidatos a servir de corrección, para cada una de las palabras leídas por el dispositivo, basándose para ello en probabilidades de confusión, y en un diccionario basado en *tries*.

(2) Juntaba aquellas parejas de palabras en las que se presumía la existencia de errores de separación de palabras.

(3) Reordenaba los candidatos, esta vez basándose en las probabilidades de los bigramas en las palabras.

Las fases dos y tres servían para retroalimentar a los análisis hechos en las fases previas.

En el mismo año de 1991, Jones y su grupo examinaron a su sistema a través de dos experimentos. En el primer experimento, entrenaron a su red con seis documentos de software, que totalizaban un conjunto de 8'170 palabras, y luego lo examinaron con un séptimo documento, el cual poseía 2'229 palabras. El porcentaje de error que el dispositivo de reconocimiento óptico obtuvo, en esta prueba, un porcentaje de error del 16.2%, y el sistema de Jones y su equipo corrigió el 89.4% de estos errores. En el segundo experimento, entrenaron al sistema con un total de treinta y tres páginas de texto noticioso publicado por A.P., y luego lo examinaron con doce páginas de la misma fuente. El porcentaje de error en el reconocimiento óptico, realizado sobre las doce páginas del examen, varió entre el 6.8% y el 10.3%, y el sistema corrigió el 72.5% de esos errores.

Independientemente de los porcentajes de exactitud que alcanzó el sistema de estos investigadores, así como del éxito obtenido en la programación del corrector, uno de los detalles más significativos del trabajo hecho por Jones, Story y Ballard, es que fue uno de los primeros en incorporar conocimiento contextual al momento de calcular las probabilidades de las palabras y los signos de puntuación dentro de los procesos de corrección.

Recientemente, las técnicas que hacen uso explícito de las probabilidades de error, han comenzado a ser explotadas para las aplicaciones de corrección ortográfica. En 1984, R. L. Kashyap y B. J. Oomen publicaron un artículo hecho bajo esta óptica (*Spelling correction using probabilistic methods*). Kashyap y Oomen intentaron explorar el uso de las probabilidades de error, como consecuencia del pobre desarrollo que las metodologías tradicionales de la corrección ortográfica, adquieren en la revisión de palabras de menos de seis caracteres de longitud. Observaron, entre otras cosas, que los errores ortográficos, involucran, frecuentemente, la escritura

probabilidad de aparición de letras individuales, y las aristas de la gráfica manifiestan las probabilidades de transición existentes entre las diferentes letras. Posteriormente, con ayuda del algoritmo de Viterbi, la gráfica es eficientemente recorrida hasta encontrar la sucesión de letras que suma la probabilidad más alta de ser una palabra correcta, dadas las estimaciones hechas sobre la lectura particular proporcionada por el dispositivo utilizado en esta tarea, y dadas las probabilidades de transición conocidas en el idioma.

En la actualidad, los anales de la investigación relativa a la corrección ortográfica automatizada, dispone de numerosas modificaciones hechas sobre el algoritmo de Viterbi, pero ninguna ha podido evitar de un modo apropiado y definitivo a la principal de las desventajas que tiene el algoritmo. En efecto, desde la primera vez en que este algoritmo fue usado en el ambiente de la corrección ortográfica, y hasta el día de hoy, pasando por todas sus modificaciones, siempre ha conservado el problema de ser capaz de generar palabras que no existen en el idioma. Recordemos que el algoritmo simplemente devuelve la cadena cuyas letras agrupadas en un orden apropiado suman la más alta probabilidad de corrección, pero una sucesión de letras, poseedora de la más alta de las probabilidades posible, no siempre coincide con una palabra válida, por ilógico que esto sea. Por esta razón es que estudiosos como Toussaint, Hull y Srihari han concluido que la exactitud de las técnicas basadas exclusivamente en las probabilidades de confusión y transición, deja mucho que desear, y la mayor parte de las veces es mediocre.

Otra técnica más, que también pretende explotar a las probabilidades de confusión y de transición sin el apoyo de ningún otro recurso más, es la conocida con el nombre de **técnica de relajación** (*relaxation technique*), descrita por A. Goshtasby y R. W. Ehrich en 1988, en su artículo *Contextual word recognition using probabilistic relaxation labeling*. Esta técnica revisa a cada cadena capturada por el dispositivo de reconocimiento óptico, de manera que las palabras que sean reportadas como leídas ya hayan sido ortográficamente revisadas. Su funcionamiento procede de la siguiente manera, una vez recibido el primer caracter de la palabra, el algoritmo termina de construir el resto de la palabra, cotejando los caracteres leídos con las estimaciones de probabilidad de aparición de cada caracter. Una fórmula conocida con el nombre de fórmula de relajación de Rosenfeld es utilizada para ajustar, iterativamente, las probabilidades de cada caracter como una función de las probabilidades de transición de los caracteres adyacentes. El número de caracteres candidatos a convertirse en parte de la palabra capturada por el dispositivo, puede ser limitado a través de cotas adecuadamente escogidas. La complejidad computacional que el algoritmo de la relajación manifiesta para cada palabra, es del orden de $10n^2$, donde n es el número de las letras de la palabra.

La mayor de las desventajas que tiene la técnica de la relajación es que, de la misma forma que el algoritmo de Viterbi, puede llegar a producir palabras inexistentes en el idioma. Tan sólo por citar un ejemplo, vale la pena mencionar que en un examen que le fue practicado a un sistema programado bajo esta metodología, la computadora convirtió a la palabra correcta *biomass* en la palabra inexistente *biomess*, debido a que la sucesión de letras constitutiva de la primera de estas palabras es menos probable que la sucesión que genera a la cadena biomess. Goshtasby y Ehrich consideran que la técnica de la relajación podría ser mejorada, si se le agrega información adicional. Por ejemplo, si el sistema es adicionado con una base de datos formada por bigramas que aporten información sobre las sílabas válidas y más probables en el idioma, entonces, durante la ejecución del algoritmo de la relajación, aumentarían, en cada iteración, las probabilidades de los caracteres que son candidatos a ser parte de la cadena leída, y que pueden formar sílabas compatibles con el idioma. También aceptaron que el proceso de relajación puede ser perfeccionado con el acompañamiento de alguna técnica basada en la utilización de diccionario, con todo y que esto va en contra del propósito original de evitar las búsquedas en diccionarios.

Otra forma de utilizar la información probabilística en las aplicaciones de corrección ortográfica, con muy buenos resultados, por cierto, es como preprocesador de correctores. La actividad en este terreno comenzó en 1988, cuando T. Oshika y otros tres investigadores a su cargo, implementaron un procedimiento de esta naturaleza, al cual denominaron **Modelo de Markov Oculto** (*Hidden Markov Model*, o bien, *HMM*, para abreviar). Este procedimiento sirvió para preclasificar apellidos de acuerdo con información étnica conocida y estudiada anteriormente. Los procedimientos *HMM* fueron hábilmente diseñados por Oshika y su equipo, uno para cada uno de los cinco diferentes idiomas involucrados con la aplicación requerida, basándose para ello en las probabilidades de transición de sucesiones de letras tomadas de muestras de 2'000 apellidos de cada lenguaje. Durante la ejecución del sistema,

los cinco procedimientos *HMM* se usaron para clasificar, automáticamente, a los nombres tecleados en la computadora, antes de determinar si es que estaba correcta o incorrectamente tecleado, para en tal caso generar sus posibles variantes ortográficas (correcciones), las cuales, por supuesto, habrían de depender específicamente del lenguaje correspondiente al nombre. Oshika reportó que su técnica de preprocesamiento, alcanzó una exactitud del orden del 69% al 88%, en un examen que le fue practicado sobre un conjunto de 160 nombres, cuya procedencia lingüística era difícil de determinar.

Por lo que a las técnicas basadas en probabilidad se refiere, el mejor desempeño en la actividad de la corrección ortográfica, ha sido obtenido por las técnicas que combinan el uso de información probabilística, información ascendente (*bottom-up information*), con la información desprendida de los diccionarios, información descendente (*top-down information*). Quizá la razón de esta peculiar característica, pueda derivarse de un comentario hecho por los estudiosos R. Shingal y G. T. Toussaint, en su artículo titulado "*A bottom-up and top-down approach to using context in text recognition*". Su comentario es el siguiente: "*los métodos que emplean diccionarios tienen porcentajes de error considerablemente bajos, pero siempre acompañados de grandes demandas de memoria, además de una alta complejidad computacional. Por su parte, los métodos de Markov tienen exactamente la característica inversa*". Luego entonces, cualquiera pensaría que una adecuada combinación de ambos podría producir la solución más adecuada. Justamente con esta idea en mente, Shingal y Toussaint inventaron una técnica que ellos mismos llamaron predicción-corrección, la cual lleva a cabo su trabajo de modo muy eficiente, gracias a su capacidad de buscar en el diccionario con la mitad del costo acostumbrado. La técnica exige una ordenación especial del diccionario, hecha conforme a un valor precalculado *V*, mismo que le es asignado a cada una de las palabras válidas del léxico, a través de sus probabilidades de transición. Este método primero utiliza una versión modificada del algoritmo de Viterbi, para poder reconocer cada una de las palabras recibidas por la máquina. Posteriormente, calcula *V* sobre la palabra reconocida, y termina haciendo en el diccionario una búsqueda binaria del vocablo con un valor semejante. Si esa palabra no es exactamente la palabra reconocida, entonces calcula *V* para las palabras ubicadas en una cierta vecindad de ella, y selecciona aquella que tenga el puntaje más alto. En un experimento hecho con un diccionario de 11'603 palabras, y dos bloques de texto, uno de 2'521 palabras y otro de 2'256 palabras, sobre los que se realizó el reconocimiento óptico, la técnica obtuvo un porcentaje de eficiencia del 96.6% y del 96.4%, respectivamente. Para mala fortuna, Shinghal y Toussaint observaron que su técnica trabaja mejor cuando la segunda búsqueda no se extendía sólo sobre una vecindad de la palabra, sino sobre todo el diccionario completo. Así pues, ellos mismos hicieron notar la necesidad de un mejor método de organización del diccionario.

También en el ambiente de las técnicas que vinculan a los diccionarios con la probabilidad, vale la pena mencionar el trabajo de S. N. Srihari, J. J. Hull y R. Choudhari, quienes en 1983 programaron una técnica en la cual el diccionario fue representado como un *trie*, y además utilizaron el algoritmo de Viterbi para manejar información probabilística. Su técnica fue examinada con un texto de 6'372 palabras y un diccionario de 1'724 palabras. Sus resultados indicaron que el algoritmo de Viterbi, por sí sólo, era capaz de corregir el 35% de los errores en los que incurrió el dispositivo de reconocimiento óptico que utilizaron, mientras que su algoritmo basado en su particular implementación del diccionario, fue capaz de corregir, por sí sólo, el 82% de los errores. finalmente, la combinación de los algoritmos de Viterbi y de la búsqueda en un diccionario almacenado como *trie*, obtuvo un porcentaje de exactitud del 87%.

En 1988, R. M. K. Sinha y B. Prasada desarrollaron una técnica que no solamente incluyó probabilidad y búsqueda en diccionarios, sino que además tuvo incorporadas una variedad de heurísticas. Su meta final era la de atacar el problema de que, en la práctica, es imposible asumir que existe un diccionario que contendrá todas las palabras que puedan llegar a encontrarse en un determinado texto. Así que ellos compilaron un "diccionario parcial" que contenía 10'000 de las palabras más frecuentes del idioma, y lo fueron complementando con todas las palabras válidas obtenidas por sustituciones sencillas de caracteres. Su algoritmo contaba con dos fases. En la primera, corregía sólo aquellos errores para los cuales sus probabilidades de confusión producían una palabra válida del diccionario parcial. Luego, en una segunda fase, el algoritmo utilizaba una versión modificada del algoritmo de Viterbi, para estimar la más probable corrección para aquellas palabras que se hallaban fuera del diccionario. El diccionario estuvo almacenado como un *trie*, y las variadas heurísticas del sistema fueron empleadas para clasificar a los candidatos a corrección, para limitar la búsqueda a través del *trie*, y también para limitar al algoritmo de

Viterbi. Esta técnica fue examinada con un texto de 5'000 palabras (25'000 caracteres en total), escritas en máquina de escribir y por otros medios, además de contar con diferentes tamaños y estilos de letra. Su porcentaje total de exactitud en el reconocimiento óptico de caracteres, excedió el 98%, y su exactitud en el reconocimiento de palabras completas, estuvo alrededor del 97%, sin contar errores de puntuación. Otras observaciones hechas sobre la base de este trabajo, fueron que el diccionario aumentado tiene un mejor desempeño en la corrección de textos no convencionales, como por ejemplo documentos literarios y técnicos, que el diccionario no aumentado. Sin embargo, en la corrección de textos convencionales, el diccionario aumentado deteriora el funcionamiento del sistema y lo conduce a porcentajes de exactitud menores al 90%.

En 1991, M. A. Jones, G. A. Story y B. W. Ballard programaron un postprocesador para el reconocimiento óptico de caracteres, el cual integró un tratamiento bayesiano de predicciones, mismas que fueron hechas a partir de una multitud de fuentes de conocimiento. En una fase de entrenamiento anterior a la utilización del sistema, Jones y los demás construyeron modelos estadísticos para el dispositivo de reconocimiento óptico de caracteres, al tiempo que también prepararon relaciones sobre el tipo de documentos que serían procesados. Sus modelos incluían información recopilada de muy variadas fuentes: las frecuencias de las letras en monogramas y bigramas, trigramas binarios y frecuencias de las letras iniciales de los diferentes n-gramas, frecuencias de los monogramas y bigramas en palabras completas y análisis sobre la puntuación y utilización de las letras mayúsculas y minúsculas.

El corrector de Jones y su equipo funcionaba en tres etapas:
(1) Generaba una lista ordenada de los posibles candidatos a servir de corrección, para cada una de las palabras leídas por el dispositivo, basándose para ello en probabilidades de confusión, y en un diccionario basado en *tries*.

(2) Juntaba aquellas parejas de palabras en las que se presumía la existencia de errores de separación de palabras.

(3) Reordenaba los candidatos, esta vez basándose en las probabilidades de los bigramas en las palabras.

Las fases dos y tres servían para retroalimentar a los análisis hechos en las fases previas.

En el mismo año de 1991, Jones y su grupo examinaron a su sistema a través de dos experimentos. En el primer experimento, entrenaron a su red con seis documentos de software, que totalizaban un conjunto de 8'170 palabras, y luego lo examinaron con un séptimo documento, el cual poseía 2'229 palabras. El porcentaje de error que el dispositivo de reconocimiento óptico obtuvo, en esta prueba, un porcentaje de error del 16.2%, y el sistema de Jones y su equipo corrigió el 89.4% de estos errores. En el segundo experimento, entrenaron al sistema con un total de treinta y tres páginas de texto noticioso publicado por A.P., y luego lo examinaron con doce páginas de la misma fuente. El porcentaje de error en el reconocimiento óptico, realizado sobre las doce páginas del examen, varió entre el 6.8% y el 10.3%, y el sistema corrigió el 72.5% de esos errores.

Independientemente de los porcentajes de exactitud que alcanzó el sistema de estos investigadores, así como del éxito obtenido en la programación del corrector, uno de los detalles más significativos del trabajo hecho por Jones, Story y Ballard, es que fue uno de los primeros en incorporar conocimiento contextual al momento de calcular las probabilidades de las palabras y los signos de puntuación dentro de los procesos de corrección.

Recientemente, las técnicas que hacen uso explícito de las probabilidades de error, han comenzado a ser explotadas para las aplicaciones de corrección ortográfica. En 1984, R. L. Kashyap y B. J. Oomen publicaron un artículo hecho bajo esta óptica (*Spelling correction using probabilistic methods*). Kashyap y Oomen intentaron explorar el uso de las probabilidades de error, como consecuencia del pobre desarrollo que las metodologías tradicionales de la corrección ortográfica, adquieren en la revisión de palabras de menos de seis caracteres de longitud. Observaron, entre otras cosas, que los errores ortográficos, involucran, frecuentemente, la escritura

errónea de letras adyacentes en el teclado, situación ya comentada por nosotros, y basados en ello, compilaron una tabla de estimaciones subjetivas de la probabilidad de tales errores de sustitución. También incluyeron estimaciones subjetivas de las probabilidades de existencia de supresiones de letras en las palabras, así como inserciones de cualquier caracter en las diversas posiciones de una palabra. La programación del sistema corrió bajo el supuesto de que todos los posibles errores ortográficos existentes, pueden reducirse a una combinación de simples sustituciones, inserciones o supresiones, de modo que idearon un algoritmo recursivo que hizo uso de sus estimaciones para calcular la probabilidad de que dada una cadena ortográficamente errónea, esta fuese una versión deformada de alguna de las palabras del diccionario.

Después de implementar su sistema, Kashyap y Oomen lo examinaron con errores ortográficos artificialmente generados, en palabras de tres, cuatro y hasta cinco caracteres de longitud. Los errores ortográficos fueron generados de acuerdo con las tablas de probabilidades de error, y basándose en un promedio, previamente establecido, de entre 1.65 y 1.90 errores ortográficos por palabra. El diccionario que emplearon, fue uno conformado por las que a su juicio fueron las 1'025 palabras más comunes en el idioma y inglés, y que como ya se dijo, tuvieran longitud mayor a dos y menor a tres. Kashyap y Oomen reportaron que su corrector tuvo un porcentaje de exactitud que varió entre el 30% y el 92%, dependiendo de la longitud de la palabra, y del promedio del número de errores por palabra que fuera considerado. Según Kashyap y Oomen, su investigación se compara favorablemente con la realizada por T. Okuda, E. Tanaka y T. Kasai en 1976, y que fuera publicada en 1976, bajo el título de *A method of correction of garbled words based on the Levenshtein metric*. El sistema de Okuda y los otros estudiosos de la corrección que trabajaron con él, promedio una eficiencia del 28% al 64%, para palabras de longitud similar a las de Kashyap y Oomen, además de que también trabajaron con errores de las mismas características. Okuda y su grupo utilizaron una técnica de mínima distancia de edición que combinó las métricas de Damerau y Levenshtein.

Quizá uno de los correctores ortográficos basados en probabilidad, que mayor popularidad ha adquirido, sobre todo por la calidad de la investigación que lo produjo, fue el realizado por M. D. Kernighan, K. W. Church y W. A. Gale en 1990. Estos tres estudiosos y reconocidos programadores, inventaron un algoritmo especializado en la corrección de errores ortográficos sencillos, al cual llamaron *correct*. Como preparación para el sistema, los tres investigadores crearon una base de datos de errores ortográficos genuinos, extraídos todos ellos de un bloque de cuarenta y cuatro millones de palabras de texto noticioso de la compañía A.P. Luego compilaron cuatro matrices de confusión, una por cada una de las cuatro clases de errores ortográficos simples, a saber, inserciones, supresiones, sustituciones y transposiciones. También estimaron probabilidades de confusión de todas las estadísticas de que dispusieron, y utilizaron una técnica inversa de distancia de mínima edición para generar un conjunto de candidatos a servir de corrección de errores sencillos, para más tarde identificar un error específico dentro de cada candidato, y continuar con un cálculo Bayesiano para ordenar a los candidatos.

Correct comenzaba con una cadena ortográficamente errónea detectada por *Spell*. Primero, generaba su conjunto de candidatos a corrección, a través de una evaluación sistemática de todas las posibles permutaciones tipográficas que dan lugar a errores sencillos, en la cadena ya de por sí errónea, verificando si es que alguna de esas permutaciones producía una palabra válida en el diccionario de *Spell*. Una tabla de supresiones calculada con antelación, así como una función de dispersión conveniente, minimizaron el tiempo de acceso al diccionario. Posteriormente, los candidatos fueron organizados en base a la multiplicación de la probabilidad *a priori*, que le era asignada a la aparición del correspondiente candidato (es decir, su frecuencia normalizada), por la probabilidad de aparición del error específico por el cual, el candidato en cuestión, difería de la cadena ortográficamente errónea.

Correct fue evaluado con un conjunto de trescientos treinta y dos errores ortográficos sencillos, hallados en texto de la *A.P.*, y que pertenecían a publicaciones hechas con un mes de diferencia con respecto de aquellas que sirvieron para alimentar de información al sistema. Cada uno de los trescientos treinta y dos errores ortográficos, poseía, exactamente, dos candidatos a servirles de corrección ortográfica. Además de *correct*, tres personas fueron también evaluadas con el mismo conjunto de errores, también con la finalidad de que hallaran la corrección precisa. A las personas examinadas, se les concedió la oportunidad de no seleccionar ninguna de las dos opciones dadas por error, y no les fue dada la oportunidad de disponer de información contextual relativa a la aparición del error

ortográfico. Como resultado del examen, los investigadores reportaron que *correct* seleccionó la misma corrección que al menos dos de las personas examinadas en el 87% de las veces.

Después de su trabajo en *correct*, Kernighan se dedicó a elaborar una versión de *correct* especial para aplicaciones de síntesis de texto a voz. Antes de pronunciar una palabra, el componente de pronunciación del sistema, debía calcular un *índice de dificultad de pronunciación*, con la finalidad de eludir correcciones innecesarias de palabras ortográficamente erróneas, pero cuya pronunciación sería aceptable, como por ejemplo, *operater* y *wud*. De este modo, un error ortográfico le era reportado al corrector, sólo cuando el *índice de dificultad* excediera un cierto límite. En un experimento, se le pidió a un grupo de personas que escucharan al sintetizador que generaba la voz que la computadora producía, luego de leer el texto que le era alimentado. Se observó que la comprensión de las palabras pronunciadas por la máquina, con ayuda de esta versión de *correct*, mejoró considerablemente, gracias al sistema implementado por Kernighan.

CAPÍTULO V
REDES NEURONALES

Al menos sobre el papel, las redes neuronales prometen constituir el más idóneo de los candidatos para el servicio de los correctores ortográficos, sobre todo si pensamos en su capacidad inherente para asimilar asociaciones, ya sean de entradas incompletas o de entradas defectuosas. Además, dado que las redes neuronales pueden ser entrenadas sobre la marcha, esto es, dado que se puede conseguir que perfeccionen su capacidad correctora, a través del contacto directo con errores ortográficos reales, entonces, pueden ser adaptadas a los patrones de error en los que específicamente incurre su comunidad de usuarios, de manera que, sin necesidad de mucho esfuerzo, sea posible maximizar la exactitud de su corrección en cada aplicación particular. Con un poco de imaginación, uno podría pensar que una red neuronal es como un estudiante de primaria que está aprendiendo ortografía, claro, no en un salón de clases, sino en una estación de trabajo personal, en donde ejecución tras ejecución, se adapta más y más a la idiosincrasia de su dueño. Desafortunadamente, esta metodología está apenas por mostrar su verdadero potencial, ya que, hoy en día, los requerimientos computacionales que exige superan, y con mucho, la capacidad y las expectativas del hardware existente. De cualquier modo, en este capítulo deseamos profetizar un poco sobre aquello que consideramos que, en el futuro, ocupará las páginas centrales de la mayoría de los artículos de la automatización ortográfica, explicando también las ideas principales de lo que ya se ha hecho con las redes neuronales, con el objeto de no quedarnos al margen en esta materia. Primeramente, iniciaremos con el algoritmo más utilizado en las tareas de corrección ortográfica para el entrenamiento de redes: el algoritmo de *propagación hacia atrás*.

§1. Propagación hacia atrás

Sin temor a equivocarnos, podemos afirmar que el verdadero inicio de la carrera de las redes neuronales, en el campo de la corrección ortográfica, se presentó en el año de 1986, cuando D. E. Rumelhart, G. E. Hinton y R. J. Williams, publicaron un libro titulado *Learning internal representations by error propagation*, en el cual describieron el diseño de un algoritmo que lleva por nombre **algoritmo de propagación hacia atrás** (*back-propagation algorithm*), y con él echaron a andar la investigación de esta nueva rama.

En la mayoría de los casos, una red de propagación hacia atrás, está compuesta por tres estratos de nodos: el primero, un estrato de ingreso, el segundo, un estrato intermedio, casi siempre llamado estrato oculto, y por

último, un estrato de salida. Cada uno de los nodos del primer estrato, o estrato de ingreso, está conectado, a través de una liga con peso asignado, a cada uno de los nodos del estrato oculto. Análogamente, cada uno de los nodos del estrato oculto está conectado, por medio de una liga con peso asignado, a todos y cada uno de los nodos del estrato final. La información de entrada, así como la de salida, es representada, respectivamente, en los nodos de entrada y de salida de la red, a través de *patrones de actividad*, los cuales pueden estar encendidos o apagados, dependiendo de la intención que pretenda señalarse. Por ejemplo, para indicar que un cierto nodo está encendido, se acostumbra, por convención, asignarle un uno, mientras que para indicar que está apagado, comúnmente se le hace corresponder un cero. En ocasiones, se desea manejar información mucho más precisa, y el simple hecho de que un nodo esté apagado o prendido, no basta para procesar una respuesta adecuada, razón por la que en vez de ceros y unos, los nodos son caracterizados con números reales, de modo que proporcionen una cuantificación más exacta del nivel de actividad existente en cada uno de ellos.

El flujo de los datos a lo largo de una red de propagación hacia atrás, comienza con la colocación de un patrón de actividad en los nodos de entrada, el cual transmite la corriente de actividad a todas las ligas con peso asignado. Cuando la corriente ha conseguido llegar al nivel los nodos ocultos, un patrón oculto es calculado, antes de que la actividad fluya a los nodos de salida, en donde un patrón más, esta vez un patrón de salida, es también calculado. Los pesos de las ligas, representan la intensidad de la conexión entre los diferentes nodos, además de que sirven como resistencias, pues modifican la cantidad de actividad alcanzada en un nodo de nivel superior, desde un nodo de nivel inferior. La actividad total de un nodo, se define como la suma de las actividades de cada uno de los nodos de nivel inferior que llegan a él, multiplicada por la intensidad de su conexión. En la mayoría de los sistemas de propagación hacia atrás, la suma de las actividades de los nodos inferiores es "arreglada", de modo tal que produzca siempre un valor necesariamente igual a uno o a cero. Otras veces, la suma es "corregida", con la ayuda de una función que la obliga a devolver un valor real, entre el 0.1 y el 0.9.

El algoritmo de propagación hacia atrás, proporciona una técnica para determinar el conjunto de los pesos de las ligas de la red, mismos que permitirán generar, dentro de cada aplicación particular, el patrón de salida más correcto para todo patrón de entrada dado, o en el peor de los casos, el patrón más cercano al correcto. Este algoritmo es, en realidad, una técnica de convergencia, por cuanto se espera que los pesos converjan a un valor que optimice los resultados, y se fundamenta, poderosamente, en un entrenamiento a través de ejemplos. Así pues, cuando la red apenas ha sido programada, las ligas poseen pesos pequeños, que les son asignados aleatoriamente durante el proceso de inicialización del sistema, pero que después, serán ajustados de acuerdo con el resultado desprendido de cada pareja formada por una entrada, y por su respectiva salida, todas ellas tomadas de un seleccionado conjunto de entrenamiento. Más en detalle, el proceso se efectúa como sigue: dada una entrada propia del conjunto de entrenamiento, esta es colocada en los nodos de entrada de la red, para que la activación sea enviada, a través de las ligas con peso, hacia los nodos ocultos, y continuar así hasta producir un patrón de salida en los nodos finales. El patrón real de salida es entonces comparado con el patrón de salida deseado, para calcular una diferencia en cada uno de los nodos del estrato de salida, diferencia basada en operaciones ideadas con anterioridad. Después de esto, se inicia en la red el trabajo de regreso, o de "propagación hacia atrás". La diferencia obtenida en el paso precedente, es utilizada para ajustar cada uno de los pesos de las ligas de la red. A simple vista, el ajuste no es extraordinariamente notorio, pues sólo se modifica al peso en una pequeña cantidad, inversamente proporcional al error existente, pero con eso es más que suficiente para que el algoritmo ahora sí produzca el resultado querido. Todo esta rutina se repite una y otra vez, hasta que todas las palabras presentes en el conjunto de entrenamiento hallan sido tomadas en cuenta, y hasta que todos los pesos converjan, es decir, hasta que las ligas de la red cuenten con los pesos adecuados para producir, siempre, un patrón de salida muy cercano al correcto, para todos los patrones de entrada del conjunto de entrenamiento. Posteriormente, los mismos pesos así obtenidos, son utilizados para corregir palabras distintas de aquellas que sirvieron de prueba, y por eso es que se dice que las redes tienen la capacidad de generalizar sus resultados.

Coloquémonos en el caso particular de que una red de propagación hacia atrás, halla sido programada para una aplicación de corrección ortográfica. Entonces, cualquier error detectado, de la misma forma que cualquier cadena capturada del texto, es representado como un vector n-grama binario, y podría servir, desde ese momento, como patrón de entrada de la red. Su patrón de salida correspondiente, podría ser, por ejemplo, un vector de *m*

elementos, donde *m* es el número de palabras del diccionario, y sólo el nodo correspondiente a la palabra correcta estaría encendido. A veces, a este tipo de red se le nombra *clasificador 1 a m*, porque el objetivo del algoritmo sobre el cual descansa su programación, es maximizar la actividad en el nodo de salida, el cual representa a la palabra correcta, y minimizar la actividad en todos los demás nodos. De hecho, los valores de los nodos en el vector de salida resultante son, en ocasiones, interpretados como valores de probabilidad.

Para las aplicaciones ortográficas, el esquema de codificación *1 a m*, utilizado para la cadena de salida, es un esquema de codificación local, porque cada palabra en el diccionario es representada por un solo nodo. Conviene aquí decir que, en contraste con este esquema, está la metodología de codificación en n-gramas binarios, la cual es, más bien, un esquema de codificación distribuida, pues en ella ningún nodo posee toda la información necesaria para representar una palabra, y dicha información tiene que dispersarse a todo lo largo de los nodos de entrada de la red.

§2. Aplicaciones

Las redes neuronales están siendo intensamente requeridas en el trabajo con dispositivos de reconocimiento óptico de caracteres, no tanto para el software común en la lectura de palabras, pero sí para el software diseñado para el reconocimiento de números escritos a mano, tales como los sistemas de lectura de códigos postales y los de recepción de tarjetas de crédito y cheques bancarios. Comentarios y explicaciones relativas a estos temas, pueden ser hallados en un artículo publicado por O. Matan, C. J. C. Burges, Y. LeCun y S. S. Denker en 1992, bajo el título de *Multi-digit recognition using a space displacement neural network*, y en uno publicado también en 1992 por J. Keeler y D. E. Rumelhart, bajo el nombre de *A self-organizing integreted segmentation and recognition neural net*. Todas estas aplicaciones de las redes, involucradas con los lectores ópticos, cuentan con un inevitablemente alto nivel de dificultad, entre otras cosas debido a la escasez de información contextual disponible. A pesar de ello, los investigadores no han cesado en su intento por explotar a estas técnicas. Un ejemplo de esto fue dado en 1987, cuando D. J. Burr implementó un reconocedor de palabras impresas a mano, que funcionaba en dos etapas: durante la primera etapa, el sistema disponía de una técnica de reconocimiento basada en redes neuronales, mientras que en la segunda etapa el peso del funcionamiento recaía sobre una técnica de corrección probabilística. En la primera etapa, Burr utilizó una red neuronal con veintiséis nodos de salida, uno por cada una de las letras del alfabeto inglés, y trece nodos de entrada, por medio de los cuales capturaba los rasgos de las letras impresas a mano. La salida de esta red neuronal constituyó, en su momento, una útil distribución de probabilidad sobre las veintiséis letras del alfabeto. En la segunda fase, Burr usó la fórmula de Bayes, para estimar la probabilidad de cada una de las palabras del diccionario, utilizando para ello, las probabilidades de los caracteres individuales, así como las probabilidades de los monogramas de las palabras válidas. En un experimento posterior, Burr creó un conjunto de 208 caracteres escritos a mano, en donde se podían contar ocho apariciones por cada una de las veintiséis letras del alfabeto. Burr entrenó a su red con la mitad de ese conjunto, y la examinó con la otra mitad. La red obtuvo el 94% de exactitud en el reconocimiento de caracteres, y, a causa de este éxito, decidió examinarla contra un total de 20'000 palabras impresas a mano, pertenecientes a un diccionario. Sorprendentemente, el algoritmo de Burr consiguió un impresionante porcentaje del 99.7% de exactitud, en el reconocimiento de palabras de al menos seis caracteres de longitud.

Las aplicaciones que dentro de la automatización ortográfica han tenido las redes neuronales, son muy variadas, y cubren temas que no han sido tratados por otras técnicas de corrección. Por ejemplo, en 1990, M. Gersho y R. Reiter practicaron un estudio de validación ortográfica de los nombres de las calles, y concluyeron su labor con la edición del artículo *Information retrieval using self-organizing and heteroassociative supervised neural networks*. En el campo de la corrección necesaria durante la escritura de nombres propios, las redes han sido analizadas, básicamente, por K. Kukich, en un par de investigaciones hechas en 1988, y por V. Cherkassky y N. Vassilas, en otro par de estudios publicados en 1989. Por lo que se refiere a la síntesis de texto a voz, Kukich experimentó el uso de las redes neuronales en 1990. También se ha estudiado el desempeño de las redes en

aplicaciones hechas para el procesamiento del lenguaje natural, como se observa en un par de artículos aparecidos en 1990, escritos por los investigadores R. Deffner, K. Eder y H. Geiger. En los estudios de Kukich, Cherkassky y Vassilas, se usó el algoritmo de propagación hacia atrás para el entrenamiento de las redes, en los estudios de Gersho y Reiter, se emplearon redes de propagación hacia atrás y redes de auto organización (*self-organizing nets*), mientras que en el sistema de Deffner, Eder y Geiger, se utilizaron modelos de matrices de correlación con peso, con el objeto de combinar una multitud de fuentes de conocimiento léxico.

Para abundar en nuestra descripción sobre las técnicas de corrección ortográfica basadas en redes neuronales, lo más prudente es hacer unos cuantos comentarios relativos a una de las investigaciones hechas por K. Kukich, digamos, aquella que estuvo relacionada con la corrección de nombres propios. En dicho trabajo, Kukich, utilizó un diccionario de 183 apellidos, con los cuales entrenó a una red, común y corriente, de propagación hacia atrás. Su estrato de salida consistió de 183 nodos, uno por cada apellido del conjunto de entrenamiento, mientras que el estrato de entrada contenía 450 nodos, divididos en quince bloques secuenciales de treinta nodos cada uno, de manera que aquellos apellidos de hasta quince letras, pudieran ser representados. Cada bloque de treinta nodos, contaba con un nodo distinto por cada uno de los caracteres de un alfabeto de treinta caracteres, que fue, justamente, el alfabeto utilizado por ella. Así pues, si la letra *C*, por poner un ejemplo, aparecía como la letra inicial de una cierta cadena, el nodo que representaba a la letra *C*, quedaría encendido (es decir, se le asignaba el número uno), en el primer bloque, al tiempo que los otros veintinueve nodos restantes permanecerían apagados (es decir, con el número cero asignado). La red fue entrenada con cientos de errores ortográficos sencillos, artificialmente generados a partir de los 183 nombres antes referidos. Decenas de horas de tiempo de entrenamiento fueron requeridas para conseguir que la red convergiera, utilizando, durante todo el trabajo, un procesador equivalente al *Sun SPARCstation 2™ processor*. Posteriormente, la red todavía fue examinada con un conjunto de cadenas extra, formado por errores ortográficos sencillos generados aleatoriamente, también a partir de los 183 apellidos del diccionario. A pesar de todo, este último examen requirió tan sólo de unos cuantos segundos de tiempo de *CPU*, y luego de tanta preparación y trabajo hecha alrededor de este sistema, la red de Kukich pudo ser apreciada como uno de los más grandes éxitos de la corrección ortográfica automatizada: la red obtuvo un porcentaje de exactitud casi perfecto, de hecho, prácticamente alcanzó el ansiado 100% de eficiencia en cualquiera de los diferentes tipos de errores ortográficos, incluyendo inserciones y supresiones.

Entre algunos de los principales descubrimientos experimentales obtenidos por Kukich, podemos mencionar los siguientes:

(1) Las redes entrenadas con nombres propios ortográficamente mal escritos, aprenden mejor que las redes entrenadas con nombres bien escritos.

(2) El desempeño en la ejecución de la red se vio incrementado gracias al incremento en el número de nodos ocultos, el cual llegó hasta 183.

(3) Las redes que utilizan los esquemas de codificación local, aprenden más rápidamente que las variantes algorítmicas que emplean esquemas de codificación distribuida.

Las investigaciones hechas por Cherkassky y Vassilas en 1989, sirvieron para corroborar muchos de los descubrimientos de Kukich, aunque ellos usaron esquemas de codificación en vectores bigramas y monogramas, además del mismo esquema local que usó Kukich, es decir, dejando, en el estrato de salida, un nodo específico para cada palabra en el diccionario, de manera que sus redes también fueran clasificadores *1-a-m*. Cherkassky y Vassilas entrenaron a sus sistemas con nombres ortográficamente bien escritos, y más tarde los examinaron con un conjunto de nombres artificialmente generado, en el cual incluyeron errores de inserción y supresión, obtenidos a partir de los nombres originales. Los tamaños de sus diccionarios variaron, desde los más pequeños con veinticuatro nombres, hasta los más grandes con cien nombres. Las redes de Cherkassky y Vassilas, al igual que la de Kukich, consiguieron casi el 100% de exactitud en su ejecución, cuando trabajaron con los diccionarios más pequeños. Sin

embargo, concluyeron que el número óptimo de unidades ocultas de las que debe disponer una red, para aplicaciones ortográficas, está muy cercano al número total de palabras del diccionario. Esto es una conclusión importantísima, no tanto por lo que nos dice de primera intención, sino por lo que se puede desprender de ella, más aún, de lo que se deduce de esta conclusión depende el hecho de considerar a las redes neuronales como una verdadera técnica de corrección, o como un simple juego dialéctico de programación.

Efectivamente, si el número óptimo de nodos en el estrato oculto debe de estar cercano al número de palabras del diccionario, entonces podríamos pensar que las redes constituyen, simplemente, una manera de darle la vuelta a las rutinas de búsqueda en diccionario, y que ninguna generalización léxica importante se lleva a cabo en sus procesos de entrenamiento. No obstante, esto no puede ser cierto, y como evidencia de ello esta el hecho de que las redes son entrenadas con errores ortográficos que ellas "nunca vieron". A pesar de esto, Cherkassky y Vassilas determinaron que como las redes no son muy sensitivas a varios parámetros de entrenamiento, y que, como aparte de todo lo que se diga, requieren de un tiempo computacional considerablemente alto para su entrenamiento, no son tan apropiadas para la corrección ortográfica como lo son, por ejemplo, los modelos de matrices de correlación. He aquí el primer punto en contra de las técnicas basadas en redes neuronales, y el por qué éstas no han sido tan bien vistas, con todo y lo bien que nosotros hemos hablado de ellas.

Kukich continuó estudiando las redes neuronales en el año de 1990, esta vez con el objeto de hallar una forma eficiente de emplear a las redes en aplicaciones de síntesis de texto a voz, para las cuales necesitaría un diccionario mucho más grande, además de que debería de enfrentarse con una significativa porción de errores ortográficos de calidad múltiple. Para esta tarea, Kukich utilizó la arquitectura común de una red de propagación hacia atrás, con sus tres estratos tradicionales. En el estrato de entrada empleó 420 elementos, con información recopilada de vectores monogramas y bigramas. El estrato oculto de la red contuvo 500 nodos, y en el estrato de salida se usaron 1'142 nodos. Para fines de entrenamiento, la investigadora tuvo que recurrir a errores ortográficos generados aleatoriamente, porque no hubo suficientes errores genuinos disponibles, aunque para fines de evaluación, trabajó con 170 errores genuinos, mismos que ya había utilizado con anterioridad en otras pruebas para sistemas de corrección. En esta ocasión, la red de Kukich no alcanzó un porcentaje excelente de corrección, apenas llegó al 75%, pero la verdad es que se trataba de una labor considerablemente más complicada que las otras que había realizado. No obstante, lo que sí se podría pensar que fue una nota desagradable, es que el tiempo de entrenamiento de la red fue muy alto, ya que ocupó muchos cientos de horas de *CPU* en una *Sun SPARCstation*. Por lo tanto, la conclusión de este trabajo de Kukich, fue que las exageradas demandas computacionales del entrenamiento de una red de propagación hacia atrás, clasificador *1-de-m*, convierten a esta metodología en altamente ineficiente para fines prácticos, al menos con la potencia computacional disponible hasta la fecha.

Técnicas encaminadas a reducir el tiempo de entrenamiento, por medio de una estrategia que involucra la preparación de particiones del diccionario, ya están siendo estudiadas, al mismo tiempo que se han ido creando nuevas metodologías, que pretenden explotar arquitecturas de red alternativas. Una de estas nuevas metodologías, fue diseñada por M. Gersho y R. Reiter en 1990, y aparece publicada en su ya citado artículo *Information retrieval using self-organizing and heteroassociative supervised neural networks*. En resumen, el sistema hecho por Gersho y Reiter, estaba preparado para revisar la ortografía de una base de datos que involucraba un total de mil nombres de calles. Para llevar a buen término sus propósitos, este par de investigadores elaboró una metodología que combinaba una red de auto organización tipo Kohonen (*self-organizing Kohonen-type network*), con muchas redes pequeñas de propagación hacia atrás. Las entradas recibidas por el sistema eran, primeramente, conducidas a lo largo de su red tipo Kohonen, y luego de pasar por ella, eran transmitidas a una apropiada red de propagación hacia atrás, en la cual se verificaba un proceso de refinamiento de la corrección ortográfica. Para una base de datos de tan sólo ochocientos nombres de calles, el sistema de Gersho y de Reiter reportó una exactitud del orden del 74% al 97%, pero continúan haciendo investigaciones con bases de mayor tamaño.

§3. Futuras direcciones

Tal y como señalamos al principio de este capítulo, existen una y mil razones para suponer que las técnicas de redes neuronales, constituyen el más prometedor de los futuros para la corrección ortográfica automatizada. Nuestra suposición se basa, entre otras cosas, en el hecho de que detrás del perfeccionamiento de las metodologías de red, se halla la innegable lucha por el triunfo de la carrera tecnológica, la cual ha sido emprendida, desde hace varios años, por los fabricantes de computadoras. Como resultado manifiesto de esta lucha, podemos observar, día con día, la producción de máquinas cada vez más veloces, y con mucha más memoria de la que tenían hasta hace algunos años, de donde se deduce que, dentro de poco tiempo, el hardware requerido para las redes, mismo que a la fecha es parcialmente inexistente, estará al alcance de los investigadores, permitiendo que esta técnica deje de ser inconveniente a causa de sus excesivas demandas computacionales.

Otra de las razones que no favorece la implementación de redes en la actualidad, pero que muy pronto dejará de ser un obstáculo, es la falta de disponibilidad de grandes cuerpos de texto, los cuales son indispensables para su entrenamiento y posteriores pruebas. En relación con este problema, la *Association for computational linguistics* tomó cartas en el asunto al principio de esta década, y trabaja en la preparación de un bien dotado banco de textos, con el objeto de formar una "inmensa masa crítica" de información literal, la cual estará a la orden de quien la requiera para sus experimentos. Por conveniencia, se ha decidido que el banco de textos posea toda clase de documentación, a saber, diccionarios, redacciones bilingües, transcripciones de texto conversacional, y cualquier otro tipo de material que sea posible vincular con la actividad ortográfica.

Para muchos investigadores, una vez que hallan sido cubiertas las necesidades de hardware y de texto, las condiciones óptimas para el desarrollo de metodologías de red estarán satisfechas. La mayoría de los estudiosos considera que la mejor técnica de corrección ortográfica que podrá funcionar bajo el régimen de las redes neuronales, tendrá que ser un híbrido, con toda seguridad derivado de la combinación de las redes con la probabilidad y la estadística. Dicha combinación será ideal para distinguir a los errores genuinos de las falsas alarmas, pues a la par de que las redes proporcionarían las ventajas de un entrenamiento del sistema, la estadística y la probabilidad proporcionarían las fuentes informáticas necesarias para que la técnica nunca se despegara de la realidad. Así pues, se piensa que el corrector ideal basado en redes neuronales, debería poseer, como componente central, una unidad de procesamiento léxico, cuyo trabajo consistiría en relacionar, inteligentemente, las estimaciones probabilísticas sobre la identidad de una palabra como elemento de un texto de entrada. Los requerimientos necesarios para llevar a cabo esta tarea, llegarían al sistema desde tres diferentes fuentes: la entrada misma, la retroalimentación de una unidad de procesamiento sintáctico, y la retroalimentación de una unidad de procesamiento semántico. Durante la primera etapa de trabajo, una cadena de texto, la cual podría ser una palabra correcta, un error de palabra aislada o un error de contexto, sería transmitida hacia un detector de errores en palabras aisladas, momento en el cual, luego de unas cuantas subrutinas, tendría lugar la formación de una lista de palabras, ordenadas por probabilidad, que servirían de candidatos a ser la palabra correcta. En esta lista, se sintetizaría, a través del reconocimiento de estadísticas de error anteriormente recabadas, información sobre similitudes ortográficas, fonéticas y visuales, entre la cadena recibida y los vocablos de un gran léxico correspondiente al dominio del tema.

En una segunda fase de trabajo del corrector en cuestión, la salida desprendida del detector de errores de palabras aisladas, se convertiría en una nueva entrada, pero esta vez direccionada hacia un nuevo dispositivo corrector. En efecto, la salida de la unidad de procesamiento léxico, la cual, como ya dijimos, no es otra cosa sino una lista de candidatos ordenada por probabilidad, sería enviada, simultáneamente, a las unidades de procesamiento sintáctico y de procesamiento semántico. Este par de unidades, representaría el pilar fundamental de la detección de errores dependientes de contexto. La entrada de la unidad de procesamiento sintáctico, se transportaría hacia una subrutina de organización probabilística, que arrojaría como resultado una nueva lista de candidatos a corrección. Una potencial falta de ortografía sería detectada, si el vocablo que esta unidad dejara ubicado en la parte más alta de la lista, fuera diferente del candidato que aparecía allí antes de entrar en ella. Un procedimiento análogo se efectuaría en la unidad de procesamiento semántico, en donde se ubicaría un intérprete semántico compilado a partir

de estadísticas léxicas. El intérprete también dispondría de modelos globales y locales de léxico, así como de un modelo pragmático, que reflejaría planes y objetivos propios del sistema y del usuario. Al igual que la unidad de procesamiento sintáctico, la de procesamiento semántico organizaría a sus candidatos por medio del cálculo de probabilidades, y, una vez más, los errores se reflejarían a través de cambios en la posición más alta de la lista de candidatos. Las salidas de las unidades sintácticas y semánticas, serían retroalimentadas a la unidad de procesamiento léxico, para ser recombinadas con la lista que devolvió el detector de errores en palabras aisladas. Otra vez, la nueva lista obtenida de la unidad léxica, se direccionaría hacia las unidades sintácticas y semánticas, para que el mismo proceso antes descrito sea aplicado por segunda vez, regresando una nueva lista dirigida de vuelta hacia la unidad léxica. Este ir y venir de una unidad a otra, se repetiría tantas veces como fuera necesario para que las tres unidades mantuvieran, sin ningún cambio, al candidato alojado en la primera posición. Obviamente, se tendrían que implementar algunos procedimientos adicionales, para evitar que el sistema ingresará en un bucle (*loop*) sin salida, convirtiendo a esta metodología en una pintura de Escher.

Por supuesto, el sistema hasta aquí descrito, no es más que una conceptualización lógica de una posible solución al problema de la corrección ortográfica, pero desde luego que no tiene por qué ser la mejor. En este sentido, hay quienes consideran que las redes neuronales y la estadística, no deberían de ser las únicas metodologías involucradas en el diseño de los futuros sistemas ortográficos, aunque la mayoría coincide en que sí deberían de ser la base. No obstante, la variedad de planteamientos es muy grande, e incluso sobre el mismo modelo que acabamos de mostrar, han surgido numerosas arquitecturas alternativas, en las que se sugiere que los componentes individuales del sistema, sean implementados por separado, utilizando, cada uno, diferentes técnicas. Por ejemplo, para otro grupo de investigadores, una combinación de las métodos basadas en reglas, con los principios del cálculo de funciones estocásticas, podría ser muy útil para la programación de la rutina de reordenación probabilística de la unidad de procesamiento sintáctico, mientras que el trabajo con redes neuronales, sería ideal, algún día, para la unidad de procesamiento léxico. Por otra parte, diversos problemas de la implementación práctica de este sistema, podrían generar, por sí solos, interesantes trabajos de investigación, y aunque en este momento, este modelo constituya un vuelo de la imaginación, el futuro de la automatización ortográfica se sostiene, en gran parte, en la promesa de que la fantasía de los estudiosos produzca, para beneficio práctico, soluciones creativas.

CAPÍTULO VI
LA BÚSQUEDA DE LA PERFECCIÓN

De todo lo dicho hasta este momento, debe de resultar claro, para todos los lectores de este trabajo, que el sistema ideal de corrección ortográfica independiente, o lo que en términos triunfalistas llamaríamos, el sistema perfecto, todavía no existe. Más aún, después de haber conocido, en el capítulo anterior, un proyecto de lo que a nuestro juicio sería un sistema ideal, hablaría muy bien de nuestro sentido común, el que aceptáramos que parece imposible cubrir, al mismo tiempo, y al cien por ciento, con todas y cada una de las demandas que dicho sistema incluiría. Tan sólo por citar algunas de ellas, valdría la pena decir que si tal sistema existiera, tendría que disponer de una vasta cobertura léxica, digamos de unas 100'000 palabras, cuando menos, aunque si consideramos el hecho de que en la actualidad, ya existen correctores ortográficos con diccionarios integrados hasta por 125'000 palabras, o más, y que ni siquiera así son capaces de satisfacer a todos sus usuarios, entonces estamos hablando de la necesidad de una cobertura léxica mucho más extensa de la marcada por el requerimiento mínimo. Obviamente, el corrector perfecto tendría que ser capaz de enmendar errores ortográficos sencillos, y errores ortográficos de calidad múltiple, pero, además, su operación efectiva no estaría limitada a cierto tipo de palabras, de modo que podría trabajar con textos reales, compuestos de vocablos de corta, mediana y grande longitud. Por otro lado, realizaría su corrección en tiempo real, y la primera palabra que devolviera como posible corrección ortográfica a cada error hallado, debería de estar muy cerca de la corrección exacta, sino es que ser justamente ella en más del 90% de los casos. Ahora bien, ¿qué es lo que nos dice nuestro trabajo de los cinco anteriores capítulos a este respecto? Bueno, pues, definitivamente, nos dice que entre *Alice in Wonderland* y el corrector perfecto, existe una grandísima diferencia, porque en *Alice in Wonderland*, uno descubre cada fantasía sobre otra fantasía, mientras que en el corrector perfecto es exactamente al revés... Lo poco o lo mucho que hemos conocido a través de los cinco primeros capítulos, nos indica que conseguir una sola de todas las características atribuidas al sistema ideal, ya no digamos todas, es algo que en la actualidad aún estamos deseosos de ver, y, lo verdaderamente grave de todo este asunto, es que si bien es cierto que hoy en día un sistema con tantas exigencias, todavía no mira su primera luz, también es cierto que ya existen muchas aplicaciones que así lo requieren, por ejemplo, los dispositivos convertidores de texto a voz en tiempo real.

§1. Análisis de precisión

Hasta este momento, el progreso en las técnicas de corrección ortográfica automatizada, nos permite

afirmar que, en términos generales, podemos disponer de correctores ortográficos independientes con una exactitud cercana al 80% de perfección, y de reconocedores ópticos con un nivel de exactitud próximo al 90%, para el reconocimiento de palabras aisladas. Hacemos notar que esta afirmación es válida sólo en términos muy generales, y, de hecho, valdría la pena reafirmar esa generalidad diciendo que la gran diversidad de objetivos propios de los múltiples correctores existentes, impide hacer de este juicio una verdad consumada. En la mayor parte de las ocasiones, es imposible comparar a las diferentes técnicas de corrección conocidas, debido a la carencia de parámetros netos de confrontación. Por ejemplo, el texto de prueba que sirve para examinar a una determinada técnica, no suele ser el mismo que aquel que se utiliza para el resto de las técnicas, salvo en muy contadas excepciones, lo cual deja al texto fuera de la posibilidad de ser un digno parámetro de comparación del rendimiento de las diferentes metodologías de corrección. También hay que tener en cuenta que los diccionarios empleados para una y otra técnica, casi nunca son los mismos, e incluso, en ciertos casos, distan mucho de ser siquiera parecidos. Naturalmente, tanta disparidad entre los parámetros propios de las varias metodologías, radica en que cada técnica está fincada sobre la base de diferentes aplicaciones, por lo que fue hecha pensando en muy distintos léxicos y en muy distintos tipos de errores. Así pues, tenemos correctores para errores cometidos por dispositivos de reconocimiento óptico, y correctores para errores mecanográficos generados por humanos, pero ¿cómo equiparar a uno con otro si sirven para distintos fines? De igual forma, no podemos comparar un corrector adecuado para palabras de poca longitud, con uno para palabras de gran longitud, o bien, uno para errores sencillos con otro para errores múltiples. Luego entonces, aunque las características propias de los errores correspondientes a un cierto conjunto de prueba, el tamaño del diccionario y el contenido del mismo, son detalles que impactan fuertemente sobre la exactitud de la corrección de una técnica específica, éstos no pueden ser considerados como puntos de comparación entre unas y otras metodologías de corrección. Por lo tanto, resulta muy aventurado determinar qué tan mejor puede ser una técnica sobre otra, peor aún, tal parece que una discusión relativa a la determinación de la técnica más adecuada que existe en el ambiente de la corrección ortográfica, nos conduciría a una plática tan bizantina como aquella en la que se pretende discernir quién es el mejor escritor entre Shakespeare y Cervantes. A pesar de ello, con ánimos de disponer de un patrón de referencia congruente, los estudiosos del tema cuentan con un bien documentado análisis comparativo, debido a Karen Kukich, y en el que a las diversas técnicas se les examinó con base en un mismo conjunto de errores a corregir, para ser exactos, con base en un cúmulo de 170 errores ortográficos generados por humanos, donde el 25% era de calidad múltiple, y el 63% aparecía en palabras cortas. Los resultados obtenidos de dicho análisis, nos permiten establecer una jerarquía de eficiencia para las diferentes técnicas de corrección ortográfica independiente explicadas en este trabajo, cuando menos nos concede la oportunidad de disponer de una jerarquía parcial, lo cual ya es bastante. Dicha jerarquía la mostramos a continuación,

Metodologías basadas en la mínima distancia de edición. *Técnica: grope. Porcentaje de exactitud para un léxico de 521 palabras: 64% Porcentaje de exactitud para un léxico de 1142 palabras: 62% Porcentaje de exactitud para un léxico de 1872 palabras: 60%*

Metodologías basadas en claves similares. *Técnica: Reconstrucción atómica. Porcentaje de exactitud para un léxico de 521 palabras: 80% Porcentaje de exactitud para un léxico de 1142 palabras: 78% Porcentaje de exactitud para un léxico de 1872 palabras: 75%*

Metodologías basadas en n-gramas y distancia vectorial simple. *Técnica: Aplicación de la métrica del producto interior. Porcentaje de exactitud para un léxico de 521 palabras: 58% Porcentaje de exactitud para un léxico de 1142 palabras: 54% Porcentaje de exactitud para un léxico de 1872 palabras: 52%*

Metodologías basadas en n-gramas y distancia vectorial simple. *Técnica: Aplicación de la métrica usual. Porcentaje de exactitud para un léxico de 521 palabras: 69% Porcentaje de exactitud para un*

léxico de 1142 palabras: 68% Porcentaje de exactitud para un léxico de 1872 palabras: 67%

Metodologías basadas en n-gramas y distancia vectorial simple. Técnica: Aplicación de la métrica del coseno. Porcentaje de exactitud para un léxico de 521 palabras: 76% Porcentaje de exactitud para un léxico de 1142 palabras: 75% Porcentaje de exactitud para un léxico de 1872 palabras: 74%

Metodologías basadas en n-gramas y S.V.D. Técnica: Aplicación de la métrica del coseno. Porcentaje de exactitud para un léxico de 521 palabras: 81% Porcentaje de exactitud para un léxico de 1142 palabras: 76% Porcentaje de exactitud para un léxico de 1872 palabras: 74%

Metodologías basadas en probabilidad. Técnica: Método de Kernighan, Church y Gale. Porcentaje de exactitud para un léxico de 1142 palabras: 75%

Metodologías basadas en redes neuronales. Técnica: Propagación hacia atrás. Porcentaje de exactitud para un léxico de 521 palabras: 75% Porcentaje de exactitud para un léxico de 1142 palabras: 75%

Una muy interesante apreciación sobre estos resultados, es que los diccionarios utilizados para corregir, en cualesquiera de las técnicas examinadas, nunca sobrepasaron las dos mil palabras, lo cual habla muy bien de la investigadora que los produjo, pues, con objeto de examinar, fue capaz de aprovechar mucho a partir de poco. Ahora bien, sobre los descubrimientos revelados por los porcentajes, podemos comentar que la elevada exactitud de las técnicas de distancia vectorial basadas en n-gramas, en particular aquellas que emplearon a la métrica del coseno, puede deberse a que la representación en bigramas que esta técnica utiliza, captura una mayor cantidad de información útil para estos fines, que la recabada por la representación en monogramas de las técnicas basadas en distancias mínimas de edición. También puede notarse que el algoritmo de reconstrucción atómica, consigue compilar una muy buena cantidad de datos ortográficos, lo cual es digno de mencionar, porque tal parece que, al menos sobre el papel, la exactitud de unas y otras técnicas depende, ampliamente, de la cantidad de información manipulada por el algoritmo empleado en la tarea de corrección. Obsérvese, por otra parte, que la técnica de redes neuronales, alcanza casi el mismo nivel de precisión que el propio de la mejor de las técnicas de n-gramas y distancias vectoriales, con todo y que el algoritmo de propagación hacia atrás es entrenado con errores generados artificialmente, mientras que los errores que conforman al conjunto de prueba, fueron de origen humano. Desde luego que esto último debe ser tomado en cuenta con ciertas reservas, pues el gran número de demandas computacionales requeridas por las técnicas de redes neuronales, hace que éstas sean impracticables, más que nada por el hardware existente en esta época. Como puede verificarse, la metodología probabilística examinada, no fue capaz de superar, con notoriedad, al algoritmo que utiliza la métrica del coseno, aunque vale la pena notar que el método de Kernighan, Church y Gale, fue hecho pensando en estimaciones obtenidas de una fuente general de textos, como son los cables noticiosos de la cadena *Associated Press*, mientras que el conjunto de palabras de prueba en el examen de Kukich, perteneció a un dominio más especifico.

No está claro si es que la exactitud de los algoritmos examinados en este estudio, va acercándose, poco a poco, a una cota superior teóricamente establecida. Sin embargo, un detalle curioso es que el mismo archivo que sirvió de prueba para todos estos algoritmos, también fue usado en un examen aplicado a un grupo de personas, y por impresionante que parezca, tampoco los seres humanos fueron capaces de llegar al anhelado cien por ciento de exactitud, más aún, su precisión en la corrección ortográfica, estuvo apenas cercana al porcentaje alcanzado por algunas de las mejores técnicas computacionales analizadas. En efecto, las personas que participaron en esa prueba, obtuvieron porcentajes de exactitud oscilantes entre el 65% y el 83%, para promediar un total de 74% de eficacia. Esto no significa, en modo alguno, que los sistemas de corrección ortográfica ya hayan conseguido superar a la capacidad de apreciación humana, pero sí es digno de toda consideración el percatarse de que seis de los ocho

algoritmos estudiados por Kukich, obtuvo un mejor porcentaje, en todos los tamaños de diccionario, que el peor de los seres humanos examinados, además de que los otros dos algoritmos no estuvieron muy lejos de igualarlo. En estas condiciones, podemos garantizar que la automatización ortográfica, ya está cumpliendo con una primera intención: **realizar el trabajo mejor de lo que podría hacerse sin ayuda de una computadora.**

§2. Perspectivas

Quizás el siguiente paso que se nos antoja dar, luego de ver el anterior análisis de precisión, consiste en determinar por qué, tanto a las personas como a los algoritmos, les fue imposible llegar al cien por ciento de exactitud en la corrección del texto empleado para el examen, y averiguar si tal vez con otro tipo de texto sí se alcanzaría el deseado porcentaje. Nuestra opinión personal a este respecto, emitida sin pretender profundizar demasiado en este asunto, es que nadie pudo realizar una corrección perfecta, porque el conjunto de errores contenidos en el archivo que sirvió de examen, posee faltas cuya corrección exacta es imposible de adivinar, al menos cuando las palabras son juzgadas de modo independiente, y sin poder basarse en el contexto en el que se hallan. Por ejemplo, dado el error ortográfico *vver*, es prácticamente imposible determinar si su corrección exacta es *over*, *ever* o *very*. Si conociéramos el contexto en el que se ubica *vver*, entonces la única razón para que no pudiéramos escoger su corrección adecuada, es que nuestro dominio del lenguaje fuera demasiado escaso, pero sin saber cuál es el sentido del enunciado en el apareció *vver*, escoger su corrección ideal se convierte en un albur. La misma situación se presenta con cada uno de los siguientes errores y las posibles correcciones mencionadas:

Error ortográfico	Conjunto de correcciones
ater	*after, later, ate, alter*
wekk	*week, well, weak*
gharge	*charge, garage, garbage*
throught	*through, thought, thoughout*
thre	*there, three, threw, the*
oer	*per, or, over, her, ore*

Obsérvese que, en efecto, si miramos aisladamente a cada uno de estos errores ortográficos, o a cualesquiera que se les parezca, es muy difícil ordenar a sus posibles candidatos a corrección, de hecho, si nuestro algoritmo de ordenación estuviera basado únicamente en similitud ortográfica, la guerra estaría perdida. Por tal motivo, para fines de evaluación de metodologías, muchos investigadores afirman que, si para cada error detectado en un texto, el algoritmo empleado en su corrección es capaz de determinar un pequeño conjunto de palabras, entre las cuales se cuente a la corrección exacta, en más del 90% de los casos, entonces el algoritmo en cuestión es perfecto. La razón es que si se consigue "encerrar" a la corrección precisa dentro un pequeño conjunto, el uso posterior de técnicas de corrección ortográfica dependientes de contexto, podría determinar cuál de las palabras pertenecientes al conjunto, se debe aceptar como la corrección exacta. No discutiremos aquí ese tipo de desarrollos de programación, pero la verdad es que hay mucho de que hablar al respecto. Por otro lado, si el corrector que se desea obtener, será después integrado en alguna aplicación similar a las de los procesadores de texto, entonces basta

con obtener un conjunto de posibles candidatos a servir de corrección, para luego dejar que sea el propio usuario quien decida cuál de ellas es la más apropiada.

Varios investigadores involucrados en la automatización ortográfica, piensan que los niveles de exactitud hasta hoy alcanzados, y mostrados en estudios como el que aquí hemos exhibido, podrían ser convenientemente superados con ayuda de buenas fuentes de información probabilística, sin que esto signifique que las diversas técnicas terminen basándose más en la probabilidad que en su fuente original. Sobre el particular, nosotros ya hemos comentado que en los dispositivos de reconocimiento óptico de caracteres, las probabilidades de confusión y de transición, proporcionan una capacidad de corrección muy mediocre, siempre que son utilizadas solas, pero que mejoran su nivel cuando se les combina con técnicas de búsqueda en diccionarios. En principio, esta aseveración parece marcar el derrotero por el cual habrán de moverse en el futuro los estudiosos de esta materia, tanto en su trabajo con dispositivos lectores, como en su labor de corrección de textos producidos por personas. No obstante, esto no parece agradarle mucho a los investigadores, pues con toda seguridad implica caminar sobre abrojos, ya que establecer las probabilidades de confusión para errores generados por seres humanos, es más complicado que recabar información estadística para uso de los reconocedores ópticos. En cierto sentido, evaluar a un dispositivo lector es fácil, dado que basta con alimentarlo con texto y luego tabular la frecuencia de cada una de sus fallas, pero para el caso de los seres humanos, la simple variedad de sus faltas de ortografía dista mucho de ser reducida a pequeño número de casos, y esto implica, con toda seguridad, que, entre otras cosas, no será posible obtener una única tabla de probabilidad para todos los tipos de errores humanos.

Una de las metodologías que va a resultar favorecida con la ayuda de información probabilística, es la técnica basada en n-gramas y distancias vectoriales. Las probabilidades de error podrían, en dicha técnica, integrarse a través del almacenamiento directo de sus valores dentro de vectores que representen palabras, creando, de esta forma, modelos detallados de los vocablos, en los cuales sean caracterizadas, explícitamente, las probabilidades específicas de teclear cualquier n-grama en cualquier posición de una palabra correcta.

Por lo que se refiere a los problemas de la separación de palabras, y al de los neologismos (palabras nuevas que surgen de la creatividad morfológica y del influjo de los nombres propios), lo único que podemos decir con sensatez, es que todavía es necesaria mucha investigación, antes de llegar a obtener resultados de validez fehaciente. Como R.M.K. Sinha y B. Prasada mencionan en su artículo *Visual text recognition through contextual processing*, ningún diccionario se puede considerar completo, precisamente a causa de la creatividad morfológica a la que se prestan la mayoría de los idiomas. Su técnica, la cual se basa en la utilización de probabilidades de transición cuando detecta términos de esta naturaleza, demostró ser muy apropiada, aunque, como ellos mismos lo indican, solamente abre las puertas para futuras y más documentadas investigaciones. Otras técnicas dignas de posterior estudio, son aquellas que modelan conocimientos basados en la psicología humana, con el objeto de perfeccionar los niveles de corrección ortográfica. Tales técnicas pretenden, por ejemplo, capturar los rasgos gestalt de las palabras, esto es, calculan, entre otras cosas, el número de caracteres que se extienden por encima y por debajo de una línea horizontal de base. Por lo pronto, sólo se puede vaticinar que existe un abundante terreno que explotar en el campo de la psicología experimental, vinculada claro con la corrección ortográfica automatizada, y que trabajos como los de J. L. McClelland y J. C. Johnston, publicados en 1980, y los de J. L. McClelland y D. E. Rumelhart publicados en 1981 y 1982, podrían servir de punto de partida para esta apasionante investigación.

Es así como terminamos este sexto capítulo, concluyendo, de esta forma, la etapa documental del trabajo. Los siguientes capítulos estarán basados en lo que nuestra tarea sobre las computadoras produjo, y en las conclusiones finales.

CAPÍTULO VII
IMPLEMENTACIÓN

En todo lo que hasta el momento hemos escrito sobre el tema de la corrección ortográfica automatizada, nunca, en ningún momento, hemos establecido cuál será nuestro método a seguir en el intento por revisar la ortografía de un texto, y, en su oportunidad, en nuestro intento por corregirla. Pero si ha sido así, es porque a lo largo de los capítulos anteriores, sólo nos habíamos propuesto justamente eso: presentar una investigación completa sobre la automatización de los procesos ortográficos, abarcando tanto como nos fuera posible, desde los estudios pioneros de la materia, hasta las más avanzadas investigaciones de los años recientes, dando lugar a la producción de lo que en otros idiomas se conocería como la descripción del "*state of the art*" de la automatización ortográfica. Definitivamente, y sin temor a equivocarnos, hemos salido triunfantes de esa etapa del trabajo, porque el estudio que presentamos es más que suficiente para que ahora sí nos podamos enfrentar al problema medular de este estudio: programar un sistema de corrección ortográfica de la lengua española, capaz de detectar errores ortográficos en general, pero que preste particular atención sobre las posibles fallas en aquellas parejas de vocablos cuya ortografía difiere entre sí, tan sólo por una acento, una doble "*l*", una "*h*" muda o una "*y*", y que constituyen lo que aquí hemos llamado, las ambigüedades del español.

Una vieja anécdota del genio de la música Wolfang A. Mozart, cuenta que, alguna vez, uno de sus admiradores, deseoso de imitarlo, le pregunto qué pasos debería de seguir para componer una bella obra musical, a lo que Mozart respondió, amablemente, con una serie de ideas que a su juicio consideró pertinentes. Más tarde, cuando el admirador le preguntó a Mozart quién le había enseñado eso mismo a él desde tan pequeño, Mozart le contestó que nadie, porque de pequeño el nunca necesitó de preguntarle a nadie cómo componer. Definitivamente, la programación de un sistema de cómputo es una indiscutible fuente de anécdotas similares a esta, y es que la programación de sistemas es un arte que no sólo depende de un buen trabajo de investigación bibliográfica, sino que también depende, entre otras cosas, de la habilidad personal que cada quien tenga, de la capacidad que el programador demuestre para poder concentrar su atención en un detalle, y de su conocimiento de cómo aprovechar los instrumentos disponibles de la manera más eficiente. Dicho en otros términos, la programación es una actividad práctica, que sólo se puede dominar mediante la experiencia, pero sobre la cual no podríamos estar discutiendo sin antes habernos dedicado a estudiar, como lo hicimos en los otros cinco capítulos. De la misma manera en que un alfarero debe conocer sus materiales, comprender los principios del barnizado y del cocido, y luego aprender por experiencia, así también el programador de sistemas ortográficos, debe comprender primero la teoría, utilizar la experiencia de otros para reconocer principios generales, y después adquirir práctica. Para nosotros, ahora que ya estamos en capacidad de preparar nuestro propio sistema de corrección ortográfica, tendremos la oportunidad de transmitir también nuestra experiencia en el campo de la programación, y no sólo en el de la investigación. Quizá

tengamos que aceptar, desde un principio, que nuestra experiencia es poca, o que cuando menos nunca llegará a ser tanta como la que nos gustaría que fuera, pero por poca que esta sea, no deja de ser muy entusiasta.

Obviamente, si se desea diseñar un sistema, de cualquier índole que sea y de cualesquiera que sean las perspectivas de crecimiento que para él se tengan, tarde o temprano, se tendrá que necesitar un lenguaje para poder implementarlo, y es por ahí por donde comenzaremos este capítulo.

§1. Plataforma de desarrollo

La buena programación, así como la producción de software confiable y fácil de mantener, es un proceso independiente del lenguaje que se use. Los lenguajes de alto nivel, como *Ada* o *Pascal*, solamente simplifican el proceso de convertir un diseño en una aplicación, pero no hay ninguna razón para pensar que no se puede desarrollar buen software en cualquier otro lenguaje. Baste considerar para ello, que un buen programa bien escrito en *FORTRAN*, puede llegar a ser más legible que un buen programa mal escrito en *Pascal*. Dicho en otros términos, la legibilidad de los programas no depende únicamente de las características del lenguaje en que fueron hechos, sino también de un esfuerzo bien encaminado por parte de sus programadores, lo cual significa que incluso un programa hecho en ensamblador, se puede escribir en forma comprensible y clara.

El lenguaje que nosotros hemos elegido como plataforma de desarrollo, para la posterior implementación de nuestro sistema de detección ortográfica, es el lenguaje C, entre otras cosas porque dicho lenguaje proporciona una gran variedad de prestaciones, la mayor parte de las cuales les serán familiares a quienes hayan trabajado alguna vez en un lenguaje del tipo de *ALGOL*, como por ejemplo, el lenguaje *Pascal*. Otra razón por la que hemos escogido a C como ambiente de trabajo, es porque C fue creado, influenciado y probado en vivo, por programadores profesionales, y para beneficio de ellos. Sorprendentemente, no todos los lenguajes de programación están hechos para programadores, por ejemplo, *COBOL* y *BASIC* son dos clásicos ejemplos de lenguajes hechos para no programadores. Por su parte, *COBOL* fue diseñado para permitir que los no programadores pudieran leer y, presumiblemente, aunque improbablemente, comprender un programa. Por otro lado, *BASIC* fue creado, esencialmente, para permitirle a los no programadores programar una computadora, con la intención de resolver problemas relativamente sencillos. Caso completamente opuesto es el del lenguaje C, pues éste fue hecho para beneficio de los programadores, razón por la que C tiene todo lo que un programador sensato desea: pocas restricciones, muchas estructuras de bloque, funciones independientes y un compacto conjunto de palabras clave. Con la ayuda de C, un programador puede casi alcanzar la eficiencia del código de ensamblador, junto con la estructuración de *ALGOL* o de *Modula-2*. Por algo tenía que ser C el lenguaje más popular entre los programadores profesionales de élite, y también por eso nos decidimos a planear nuestro trabajo con C como plataforma de desarrollo.

Una de las propiedades más destacadas del lenguaje C, es que uno de los mejores y más socorridos sistemas operativos, el sistema operativo *UNIX* , está escrito, en buena parte, en lenguaje C, de aquí que *UNIX* tenga una relativa facilidad para ser instalado en máquinas distintas de la *PDP-11*, para la cual fue creado. ¿Cómo puede beneficiarle esto a nuestro programa? Bueno, al menos en principio, no tenemos ni la más mínima pretensión de generar ejecutables para *UNIX*, antes al contrario, lo que queremos es desarrollarnos en el entorno de *DOS*, sin embargo, la ventaja de hacer nuestro sistema en C, es que queda abierta la posibilidad de transportarlo a *UNIX*, en el momento en que queramos.

Por lo que a la planeación de nuestro sistema de detección ortográfica se refiere, la discusión se puede desarrollar al margen del lenguaje de programación propio del proceso de implementación, con el objeto de no afectar el entendimiento de las ideas en aquellos lectores carentes de conocimientos de C. Luego entonces, a partir de este momento, y hasta el final del trabajo, no volveremos a mencionar referencias específicas sobre este lenguaje,

salvo que consideremos que la comprensión de nuestros lectores mejoraría si así lo hiciéramos. Ahora bien, regresando a la planeación del sistema al que pretendemos dar origen, ha llegado la ocasión de hablar de la técnica que, de acuerdo con nuestra opinión, debemos de seguir para tal efecto. Por lo demás, reiteramos que la programación de un detector independiente de contexto, no es en absoluto una tarea repetitiva. Naturalmente, a estas alturas de avance tecnológico en el área de la corrección ortográfica automatizada, se han hecho ya muchos trabajos de detección independiente de contexto, pero la mayoría de ellos han sido pensados para el idioma inglés, y los pocos trabajos diseñados exclusivamente para la lengua española, han sido a expensas de grandes compañías de software, más bien interesadas en la comercialización de sus productos, y no en el progreso de una área académica.

§2. Selección de una técnica de detección

Como técnica de detección ortográfica independiente, escogimos el método de búsqueda en diccionario, por considerarlo el más apropiado para los fines de este trabajo, principalmente, porque esto nos obligará a preparar un buen léxico de propósito general, para aplicaciones de detección y corrección ortográfica, lo cual será de gran beneficio para futuras investigaciones en este ramo, pues hoy en día no existe suficiente disponibilidad sobre este tipo de herramientas. También tendremos que implementar un mecanismo de dispersión al azar, o ***hashing***, como usualmente se conoce, y esto nos capacitará para el manejo de funciones asociadas a las palabras, cosa que nos resultará muy útil, no sólo para fines de corrección ortográfica, sino también para el manejo de grandes bloques de datos almacenados en disco, sobre los cuales se requieren procesos de búsqueda, inserción y remoción de registros entremezclados unos con otros, tal y como sucederá en nuestro caso, a la vez de que se desea que los tiempos de procesamiento empleados, sean independientes del tamaño de la tabla.

El método de dispersión al azar utiliza una función a la que se acostumbra llamar ***función de dispersión***, por razones obvias, y que convierte a las llaves, que para nosotros serán palabras del idioma español, en direcciones. Específicamente, si h es una función de dispersión, y si k es una llave cualquiera, $h(k)$ es la dirección de k (dirección base de k), en una tabla a la que se denomina tabla de dispersión, y en la que se almacenará el registro cuya llave es k. Hecho esto, la recuperación del registro requerirá simplemente de calcular de nuevo la función $h(k)$. De esta forma, la inserción y la búsqueda resultan ser independientes del tamaño de la tabla. Desafortunadamente, es casi imposible evitar que la función de dispersión h, asigne la misma dirección para dos o más llaves distintas. Esta situación recibe el nombre de **colisión**, y su solución, o mejor dicho, la estrategia empleada para minimizar su aparición, ocupará buena parte de nuestros comentarios.

Alguien podría pensar que tantas descripciones teóricas en los capítulos anteriores, no sirvieron de nada si al final salimos con la simpleza de usar la búsqueda en diccionario para la programación del sistema. Sin embargo, hay que tener en cuenta que para detectar errores en texto, no hay más que dos opciones básicas: usar el diccionario o usar los n-gramas. Pero los n-gramas sólo sirven para el reconocimiento óptico, o cuando menos para eso fueron hechos, y en eso han sido primordialmente utilizados. Así pues, la única verdadera salida sensata del problema, es la de trabajar con un diccionario, por poco funcional que esto parezca. No obstante, una parte importante de este estudio, no prevista desde el comienzo, pero sí determinada sobre la marcha, fue la de realizar un análisis de los n-gramas de la lengua española. Para no interrumpir la exposición de los temas que sí tienen relación directa con la programación de nuestro detector, la parte de los n-gramas fue colocada en uno de los apéndices de este trabajo. Esperamos que nuestra presentación relativa a los n-gramas, resulte agradable para los lectores, pues para nosotros fue una de las actividades que mayormente nos satisficieron.

Ahora bien, ya una vez decidido que se tiene que trabajar con un diccionario, se debe entonces que aceptar que el uso de la dispersión al azar es uno de los mejores métodos, sin que eso signifique que dejemos de estar conscientes de las deficiencias que puede tener su empleo. Baste considerar, por ejemplo, el hecho de que los registros guardados en una tabla de dispersión, no se encuentran ordenados en ninguna forma, y más bien al contrario, parecen estar repartidos en al azar (de ahí el nombre del método), por lo que, en caso de ser necesario,

no es posible obtener los registros en orden, sin recurrir a un proceso adicional de ordenamiento. Por otra parte, como ya se indicó anteriormente, la función de dispersión puede arrojar el mismo resultado para llaves distintas, dando lugar a colisiones, de modo que una de las graves desventajas de este método, es que así como se necesita de un algoritmo para la construcción de la función de dispersión, así también se necesita de un algoritmo especial para la resolución de sus colisiones.

§3. Función de dispersión

En términos generales, podemos decir que es muy difícil determinar la función de dispersión más apropiada para una aplicación en específico. Quizá porque el sólo hecho de pretender definir el significado correcto del adjetivo "mejor", en términos de funciones de dispersión, constituye un verdadero problema. En efecto, cuando se tiene la ventaja de disponer de más de una función de dispersión, de tal manera que pueda uno darse el lujo de escoger a la mejor de ellas, se observará que es muy complicado concluir cuál de las funciones es la mejor, o cuál de ellas es la peor. Tan sólo para fines prácticos, se acepta que una buena función de dispersión, dentro de un contexto particular, es aquella que minimiza las colisiones en dicho contexto, y nada más. Así pues, aunque suene pesimista, jamás se pretenderá elaborar una función de dispersión perfecta. En la medida de lo posible, es recomendable experimentar con diversas funciones, antes de decidirse por alguna en particular. Semejantes experimentos, nos permitieron concluir que una adecuada función de dispersión para el detector de errores ortográficos, podría estar basada en una adaptación del método de la división. Según dicho método, cada registro que se integrará a la tabla, se debe dividir entre un número primo, normalmente, dado por el tamaño de la tabla, o bien, se acepta que sea el primer número primo mayor que el. Llamémosle *hash(x)* a nuestra función de dispersión, y supongamos que *cad* es una palabra del idioma español compuesta por las siguientes letras: c_1, c_2, c_3, ..., c_n, en ese orden. Hagamos *hash(cad)* igual a:

$$\text{hash(cad)} = \text{Resto}[(3 \cdot \text{ord}(c_1) + 5 \cdot \text{ord}(c_2) + \ldots + p_n \cdot \text{ord}(c_n))/TAM] \quad \ldots (*),$$

donde p_n es el n-ésimo número primo mayor o igual que tres, *TAM* es aquel número primo que se eligió como tamaño de la tabla, y *Resto(x)* es la función que devuelve el residuo de una división. Luego entonces, el simbolismo de la fórmula *(*)*, indica que el valor de la función de dispersión evaluada en *cad*, es igual al producto del valor ordinal de cada caracter de *cad*, por un número primo diferente, siempre mayor o igual que tres, y determinado por la posición que ocupa el caracter dentro de la palabra. De esta forma, una correcta implementación de nuestra función de dispersión, sugiere declarar un arreglo de números primos formado por veintidós entradas, en el cual, de manera natural, haremos que la primera entrada del arreglo corresponda al número tres, la segunda al número cinco, la tercera al siete, y así sucesivamente, hasta que la última entrada sea asignada al número ochenta y nueve. Con todas estas explicaciones en mente, no le costará trabajo al lector entender el siguiente fragmento de código, programado en lenguaje C, y que establece a nuestra función de dispersión.

```
unsigned int hash (const char *spCadena)
/* Esta es la función de dispersión */

{ /* Inicia la función hash(spCadena) */
        int i;
    unsigned int direccion=0;

  for (i=0; spCadena[i]!='\0'; i++)
  direccion+=spCadena[i]*primos[i];
    return(fmod(direccion,TAM));
} /* Fin de la función hash(spCadena) */
```

§4. Construcción y almacenamiento del diccionario

Dado que no tenemos la intención de corregir textos especializados, o escritos en la jerga técnica de ninguna materia en especial, entonces, la manera más fácil de obtener un diccionario es revisar documentos comunes y corrientes, escritos en español, claro está, y extraer de ellos todas sus palabras. Por supuesto que a ningún documento lo podemos considerar exento de errores ortográficos, ni siquiera cuando nosotros mismos lo hallamos escrito con la intención específica de luego convertirlo en diccionario. Recordemos que es práctica común, de algunos escritores de la lengua hispana, por no decir que de muchos, que en sus textos utilicen palabras que no pertenecen al idioma, y que son adecuadas para proporcionarle mayor elegancia al escrito, o simplemente, mayor expresividad. También existen textos en los que es posible hallar apariciones furtivas de vocablos incorrectos, pero que dentro del contexto propio del documento, no constituyen errores ortográficos, aunque definitivamente no se puede permitir su integración a ningún diccionario que se precie de ser bueno. Por todas estas razones, antes de distinguir a un documento con la característica de ser útil para alimentar a nuestro diccionario, primero necesitaremos depurarlo "*a mano*", como se acostumbra decir, para luego asumir la responsabilidad de considerarlo libre de errores ortográficos. Naturalmente, entre mayor sea el tamaño del documento, mayor será su utilidad para nosotros, en el sentido de que mayor será la cantidad de palabras con las que teóricamente puede alimentar a nuestro diccionario, pero también será mayor la cantidad de trabajo de depuración externa al que nos obligará. Cabe aclarar aquí, que una manera más apropiada de construir un diccionario, o cuando menos una manera más cómoda de hacerlo, sería la de disponer de todo un ejército de capturistas, encargados de copiar, letra por letra y palabra por palabra, un diccionario completo, para posteriormente guardarlo en un archivo. Sin embargo, no es ese el caso de esta investigación, simple y sencillamente porque no disponemos de tanto personal ni recursos como para darnos ese lujo, de modo que nos la tendremos que ingeniar de otra manera.

Así pues, y olvidándonos un poco de la tarea de depuración manual, nuestro algoritmo de generación del diccionario estará estructurado como sigue: recibiremos un archivo de texto, escrito en código *ASCII*, y separaremos, una por una, todas sus palabras para luego anexarlas al diccionario. Como esta técnica no favorece la obtención inmediata de todo el léxico, sino que lo construye a través de aplicaciones sucesivas, antes de integrar una palabra a la lista, será menester verificar que no esté ya presente. Ahora bien, ¿cómo iremos guardando cada una de las palabras que leamos?, o en términos más generales, ¿cómo quedará almacenado nuestro diccionario? Por comodidad y economía de espacio en disco, la tabla de dispersión que representará al vocabulario de la lengua española, se alojará en un archivo binario y secuencial, con el objeto de aprovechar las ventajas del lenguaje C. A este respecto, podemos decir que el sistema de archivos de C, está diseñado para trabajar con una amplia variedad de dispositivos, entre los que se cuentan terminales, controladores de discos y cintas magnéticas. Aunque cada dispositivo es diferente, el sistema de archivos con buffer transforma, a cada uno de ellos, en un dispositivo lógico llamado *secuencia*. Existen dos tipos de secuencias: binarias y de texto. Una secuencia binaria es una cadena de bytes en correspondencia uno a uno con los bytes del dispositivo externo, motivo por el cual, en una secuencia binaria, no se realizan conversiones de caracteres. Más aún, el número de bytes escritos, o incluso leídos, en una secuencia binaria, es el mismo que en el dispositivo externo.

Decidimos que 25'013 fuera el número máximo de entradas en nuestra tabla de dispersión, considerando la posibilidad de que nuestro diccionario almacene igual número de vocablos, y asumimos que el tamaño máximo de las cadenas de caracteres que conforman a las palabras del idioma, es 23. Claro está que un diccionario de semejantes dimensiones, nunca va a caber en la memoria de ninguna máquina, al menos no en las de tipo convencional, y es por eso que quedará guardado en disco. Sin detenernos por lo pronto en muchos detalles técnicos, diremos que la idea que tenemos en mente es alojar al diccionario en un archivo, de modo que cada una de sus líneas esté ocupada por una de las palabras pertenecientes a la lengua. Organizaremos las cosas de manera que la línea uno esté reservada para la palabra cuya dirección sea, justamente, la número uno. De manera análoga sucederá con la línea dos, y así sucesivamente, hasta colocar en la línea 25'012 a la palabra con esa misma dirección. Si por alguna razón tuviéramos la necesidad de acceder a la dirección n, posiblemente con la intención de conocer qué palabra se encuentra alojada allí, sería ideal que pudiéramos dirigirnos a ella sin antes recorrer todas las direcciones que le anteceden, y por eso es que el archivo que mantendrá al diccionario será secuencial, de modo

que nosotros sepamos, con absoluta seguridad, el número del byte que aloja al primer caracter de la *n*-ésima línea. Bajo esta misma óptica, el problema de las colisiones tiene una solución inmediata, pues bastará con disponer que la línea tenga un tamaño suficientemente grande, para poder colocar en ella a todas las palabras con igual dirección, separadas entre sí por algún caracter especial, digamos por un espacio, o bien, haciendo que cada palabra aparezca cada determinado número de caracteres.

En esencia, es así como le daremos solución al problema de guardar y accesar el diccionario, pero desde luego que tendremos que hacer algunos ajustes. Por ejemplo, tendremos que dejar de hablar de las "líneas" del archivo, pues esa terminología coloquial, sólo se acostumbra usar cuando se trabaja archivos de texto, y es bien sabido que en ninguna secuencia de ese clase, es posible llegar hasta una cierta "línea", sin primero haber recorrido a todas las anteriores. Lo que se podría hacer, en tal caso, es leer todas las "líneas", una por una, hasta obtener la que queremos, aunque por supuesto, esto nos obligaría perder mucho tiempo en el acceso a disco. Entre otras cosas, es por este motivo que abandonamos la idea de trabajar con archivos de texto, y establecimos que nuestro trabajo sería realizado con secuencias binarias. En este tipo de archivos, es factible el acceso directo a sus diferentes posiciones, puesto que dichas posiciones no son relativas a las "líneas", sino a los bytes. De esta forma, el procedimiento de construcción del diccionario, sería como sigue:

(1) Recibir el archivo de texto y abrirlo en modo lectura.

(2) Mientras que el texto no se termine, continuar con el paso número (3).

(3) Capturar la primera palabra del texto, o la siguiente palabra del texto, según sea el caso.

(4) Aplicar la función de dispersión sobre la palabra obtenida en el paso (3).

(5) Moverse a la posición que en el diccionario corresponde a la dirección antes determinada.

(6) Revisar si es que la palabra que queremos integrar, ya era parte del léxico o no.

(7) En caso de observar que la palabra leída ya estaba en el diccionario, el procedimiento es abortado y se regresa el control del programa al segundo inciso.

(8) Si se descubre que la palabra obtenida desde el paso (3) no estaba en el diccionario, entonces se procede a guardarla.

(10) Regresar al paso (2).

(11) Cerrar el archivo recién revisado y terminar.

§5. Detección de ambigüedades

La manera más natural de detectar una de las posibles ambigüedades del idioma español, es asignarle a cada palabra del diccionario un número verificador, asignado ex profeso para reconocer si es que una palabra es, o no

es, una ambigüedad. Por ejemplo, las ambigüedades pueden estar asociadas con el número uno, mientras que al resto de las palabras se les puede hacer corresponder con el cero. Físicamente, las palabras que quedarán guardadas en el diccionario, estarán acompañadas de su número verificador, el cual ocupará la posición siguiente al último de sus caracteres. El trabajo de escribir el número apropiado al final de cada palabra, conviene dejárselo a la rutina de creación y mantenimiento del diccionario. De este modo, cuando leamos un texto normal, no sólo podremos saber si las palabras que lo componen son correctas o no, sino que además sabremos si constituyen ambigüedades o no. En efecto, la detección de errores ortográficos y ambigüedades en un texto será realizada de acuerdo con el siguiente algoritmo:

(1) Recibir el texto que el usuario desea que sea revisado, y abrirlo en el modo de lectura.

(2) Mientras que el texto no se termine continuar con el paso (3).

(3) Capturar la primera palabra del texto, o la siguiente palabra del texto, según sea el caso.

(4) Aplicar la función de dispersión sobre la palabra obtenida en el paso (3).

(5) Moverse a la posición que en el diccionario corresponde a la dirección antes determinada.

(6) Leer todas las palabras que se hallan en esa posición del diccionario, hasta que se terminen o hasta que se encuentre la palabra que nos entregó el paso (3).

(7) Si la palabra del paso (3) no apareció en esa dirección, se le indicará al usuario que esa palabra no pertenece al diccionario, o bien, será inmediatamente enviada a un archivo en el que se almacenarán todos los errores ortográficos. En el primer caso, la ejecución quedará suspendida hasta el momento en el que el usuario nos indique que ha leído el mensaje que mandamos.

(8) Si la palabra del paso (3) sí fue hallada, entonces se revisará si forma parte de las ambigüedades del español o no.

(9) Si la palabra del paso (3) es una ambigüedad, se escribirá en un archivo especialmente seleccionado para el registro de ambigüedades.

(10) Regresar al paso (2).

(11) Cerrar el archivo recién revisado, el archivo para registro de errores ortográficos, en caso de que éste exista, y el archivo para registro de ambigüedades.

Para terminar con esta sección, decidimos mencionar, específicamente, algunas de las ambigüedades del idioma que tomaremos en cuenta en nuestros procesos de detección, sin que eso signifique que sean todas las ambigüedades existentes. Estas son:

el-él, porque-porqué, este-éste-esté, esta-ésta-está, que-qué, cual-cuál, si-sí, como-cómo, halla-haya, ola-hola, quien-quién, cuando-cuándo, cuanto-cuánto, cuan-cuán, tu-tú, aquel-aquél, aquella-aquélla, mi-mí, continua-continúa, proyecto-proyectó, amo-amó, ame-amé, amara-amará, amare-amaré, armo-armó, hablo-habló, canto-cantó, se-sé, saco-sacó.

CAPÍTULO VIII
CONCLUSIONES

Y así sucedió a Rabí Ismahel ben Eliša con sus discípulos, que estudiaron el libro Yĕṣirah y equivocaron los movimientos y caminaron hacia atrás, y acabaron hundiéndose ellos mismos en la tierra hasta el ombligo, por la fuerza de las letras.
(Pseudo Saadya, Comentario al Sĕfer Yĕṣirah)

Bien recordamos que al principio de este escrito, en las páginas correspondientes a la motivación, apuntamos que una de las cosas más hermosas de la investigación que en ese entonces iniciábamos, era la oportunidad de trabajar con las palabras. Eso nunca se nos olvidó, y por eso es que al término de esta labor, quisiéramos recordar especialmente uno de los artículos de Karen Kukich, en el que luego de varias páginas de exposición, cientos de referencias y cualquier cantidad de explicaciones, acabó por concluir que sin las palabras, nada hubiera sido posible, pues son ellas la materia prima de todo su trabajo. Como la misma Kukich lo dice, son *"las palabras la unidad básica de la comunicación inteligible, ... y el principal objeto de estudio de todas las actividades de procesamiento computacional de texto y voz"*. El artículo al que nos referimos es el publicado en 1992, bajo el título de *Automatically correcting words in text*. No quisiéramos hacer de este último capítulo, algo así como aquel juego en el que se trataba de llegar de salchicha a Platón en cinco pasos, por asociación de ideas. Veamos: salchicha-cerdo-cerda-pincel-manierismo-Idea-Platón. Fácil, no somos nada malos, hasta sabemos llegar de los *corn flakes* a las *matemáticas contemporáneas* en tan solo otros cinco, porque la palabra *flakes* incluye una *k*, igual que la palabra *Koenigsberg*. Koenigsberg es una ciudad con siete *puentes*, los cuales *Euler* mencionó en uno de sus conocidos teoremas, y con el dio origen a la *Teoría de las gráficas*, una de las ramas de las *matemáticas contemporáneas*. Obsérvese que dijimos que no queríamos seguir jugando, pero con todo y eso, tal parece que fue así como pasamos de la automatización ortográfica, y las técnicas de corrección, a las narraciones de las diez Sĕfirot, para después extraer un bonito parágrafo de un comentario al Sĕfer Yĕsirah, y usarlo como introducción de esta sección, en la que por si todavía fuera poca la variedad temática, empezamos por hablar de un artículo de Karen Kukich. La verdad es que no somos ningunos expertos del hebreo, y sin embargo, quisimos iniciar precisamente en estos términos, con una evocación a la *fuerza de las letras*, la que *fuera capaz de hundir a muchos hombres*, y que ojalá y que a nosotros nos enaltezca, después de haberle dedicado muchas horas de estudio y trabajo a una tarea en la que las letras, y las palabras, o como quien dice, sus posibles combinaciones, para hablar en términos de la *Témurah*, han servido de pieza fundamental. Tampoco sabemos nada de teología, y quizá seamos unos insolentes por hablar de las Sĕfirot con tanta ligereza, pero la verdad es que nos gusta mucho oír que en el Sĕfer Yĕṣirah se guarda, al mismo tiempo, *"la ciencia de la combinación de las letras y la ciencia de la purificación de los corazones"*.

***

Por mucha poesía que queramos usar en este escrito para hablar de las palabras, la realidad es que hoy en día, los sistemas de procesamiento de texto no gozan de una actualidad muy poética que digamos. Cualquiera de los conocedores del tema de los sistemas ortográficos, sabe que los dispositivos involucrados en el manejo computacional de las palabras, son indeseablemente imperfectos, y ahora también nosotros lo sabemos, pues lo pudimos corroborar durante la realización de uno de nuestros objetivos de trabajo, el de producir una antología de la automatización ortográfica. No cabe duda de que cumplimos con las expectativas que sobre esa meta fueron planteadas, según lo demuestran los anteriores ocho capítulos, viente secciones, casi cien páginas y decenas de horas de trabajo invertidas en ello. De algún modo, la sola finalización de esa antología, habría constituido un trabajo digno de ser publicado, aunque no suficiente para echar a andar la programación, puramente académica, de sistemas ortográficos del idioma español. Por lo demás, el cumplimiento de esta actividad, sirvió para documentarnos muy ampliamente en el tema de la corrección ortográfica, lo cual es de mucha utilidad, pues siempre que se expone un trabajo de investigación, por elemental que éste sea, sus autores tienen que ser los más calificados expertos en él, cuando menos por una simple obligación ética.

Otro de los objetivos de este trabajo, fue diseñar un detector ortográfico de la lengua española, y también alcanzamos ese cometido. Pareciera producto de la fantasía de uno de nuestros sueños del pasado, pero la verdad es que de la noche a la mañana, sin siquiera haberlo deseado, nos hicimos dueños de un detector de errores ortográficos hecho a nuestra medida, pues su diccionario fue creado a partir de nuestra propia expresión escrita, y por lo tanto, aunque resulte chocante, no puede existir, en ninguna parte, un mejor detector para nuestra forma de escribir que el que nosotros mismos produjimos. Claro, lo más valioso de todo esto, es que el detector que diseñamos no sólo es útil para revisar nuestros textos, sino que, eventualmente, puede usarse para cualquier otra tarea de detección ortográfica, pues basta con alimentar a su diccionario con el vocabulario conveniente. Por ejemplo, si alguna institución deseara tener un detector de faltas de ortografía, presentes en la escritura de los nombres de sus empleados, o de su clientela, bastaría con que introdujera en nuestro programa de formación de diccionarios, un archivo con los nombres propios de sus empleados, o de sus clientes, bien deletreados. Si alguien quisiera disponer de un detector de errores ortográficos, especializado en la producción literaria del castellano de principios del siglo XVII, solamente tendría que nutrir a nuestro diccionario con un texto apropiado, digamos con el del *Ingenioso Hidalgo Don Quijote de la Mancha*. Justamente, una de las principales características del desarrollo de sistemas ortográficos, es que aún cuando una buena parte del trabajo relacionado con ellos, se tenga que realizar en el ambiente académico, nunca dejarán de responder a una necesidad práctica. No por nada, las diferentes compañías fabricantes de procesadores de textos, invierten, año con año, miles de horas de laboratorio, dedicadas a reconocer los principales errores de sus usuarios, para de ese modo, satisfacer mejor las demandas de sus compradores. Como uno de los frutos de ese intenso trabajo, actualmente podemos incluso encontrar procesadores que establecen jerarquías de validez léxica, pues le permiten a los escritores establecer si prefieren que sus textos incluyan palabras tales como *translado* y *transposición*, las cuales utilizan la antigua forma de escribir el prefijo *trans*, con una *n* seguida de *s*, o si prefieren que todas esas palabras sean tachadas como errores, para que sólo se utilicen las nuevas formas *traslado* y *trasposición*, donde la *n* ha desaparecido. Este tipo de ventajas, son las que los hablantes del español más deberíamos de favorecer con nuestra atención, dado que son las que fortalecen las investigaciones de la automatización ortográfica de nuestra lengua, puesto que responden a las necesidades de corrección exclusivas del español. Un poco en ese sentido, nos encaminamos al determinar que otro de nuestros objetivos de trabajo, fuera la detección de ambigüedades. Por supuesto que para un usuario común de alguno de los tantos procesadores de textos, sería muy molesto usar un detector de ambigüedades como el que nosotros hicimos, pues la ejecución del corrector de su sistema se suspendería, una y otra vez, con la aparición de cada una de las posibles ambigüedades del idioma, pero no por ello se puede dejar de lado el deber de corregirlas. Con toda seguridad, la solución de este problema sería incluir en nuestro detector de errores, rutinas de procesamiento sintáctico, para que de ese modo, antes de advertirnos sobre la presencia de alguna ambigüedad, el sistema se asegurara de que, en efecto, está siendo mal utilizada. No obstante, el primer y más importante paso para corregir una ambigüedad es detectarla, y como quiera que sea, ese paso ya está dado.

La última de las secciones valiosas que tuvo este trabajo, y que no quisiéramos dejar sin mencionar antes de despedirnos, fue la del análisis de los n-gramas del idioma español, y que aparece en un apéndice posterior a estas páginas. Definitivamente, para nosotros, lo más significativo de esa actividad, no fue lo que hicimos, sino lo

que nos gustó hacer, lo cual es esencial para aquellos que pensamos que es inútil escribir cuando no se tiene un verdadero motivo para hacerlo. Sin lugar a dudas, el próximo paso lógico que se debería de dar, para llevar acabo un estudio más detallado sobre los n-gramas del castellano, sería construir tablas de frecuencia para cada sucesión de letras registrada, basados, por supuesto, en las cifras que obtuvimos en nuestro análisis. Esto nos ayudaría a formar arreglos n-gramas de frecuencia estadística, y con ellos podríamos realizar una primera prueba de detección ortográfica. Con toda seguridad, la sintaxis y las particularidades léxicas con las que cuenta la lengua española, favorecerán la presencia de algunas complicaciones que durante la aplicación de la técnica de los n-gramas, no son comunes a otros idiomas, y que podrían marcar la pauta de futuros trabajos. Por ejemplo, la presencia de acentos en la escritura del español, aumenta la cantidad de caracteres reales que pueden ser reconocidos en nuestro idioma, y naturalmente eso amplía, en mucho, el número de n-gramas posibles, así como el número de n-gramas válidos, por lo que se vuelve más complejo manejarlos. Precisamente en ese sentido, una magnífica investigación profesional, consistiría en medir la variación de la exactitud de una técnica de corrección basada en n-gramas, en razón del incremento, o decremento, del número de caracteres existentes en el abecedario empleado. Éste es un ejemplo, y como el debe de haber otros muchos, que se irán derivando de las nuevas complicaciones que aparezcan en el camino de otros estudios, y cuya verdadera potencialidad nunca vamos a conocer hasta que no nos enfrentemos con ellas, razón por la que fue muy importante haber comenzado como lo hicimos, y en eso reside la virtud de nuestro segundo apéndice. Es curioso, pero resulta sorprendente que todo el poder de una teoría, como la del álgebra lineal, quede conectada con este trabajo a partir de algo tan elemental, y divertido, como lo es llevar la cuenta del número de veces en que cada sucesión de letras de nuestro abecedario, aparece en las palabras del idioma. Ese tipo de cosas son las que nos permiten admirar nuestra labor con un peculiar sentido del orgullo, y que nos conceden el derecho de concluir agradecidamente esta presentación, sin llevar entre ceja y ceja la cita obligada de todos los derrotados por la vida: *Bin ich ein Gott?* Inclusive, creemos que la mejor forma de manifestar la satisfacción personal, que con certeza se desprende de la culminación de esta trabajo, es a través de una expresión que aprendimos en la filmoteca, cuando vimos la versión original de *Yankee Doodle Dandy*, con *James Cagney*: ***"We are flabbergasted!"***.

APÉNDICE 1
CRONOLOGÍA

Definitivamente, cualquiera que haya tenido la oportunidad de entrar en contacto con un trabajo de investigación, estará de acuerdo en que muchas veces, la diferencia entre un trabajo funcional y manejable, y otro simplemente útil, está dada por la organización. A propósito de este comentario, valdría la pena, en este momento, hacernos la siguiente pregunta, ¿cuáles fueron los avances tecnológicos más importantes, relativos a la automatización ortográfica, ocurridos de 1970 a 1975? Para dar una respuesta correcta a este cuestionamiento, lo de menos sería reconstruir mentalmente los acontecimientos, pero está claro que esto no es nada recomendable, pues, en nuestra mente, nunca es fácil distinguir entre los datos fiables, los erróneos y las fantasías. De mucho mayor beneficio, sería disponer de una lista ordenada de sucesos significativos para nuestro estudio, razón por la cual, desde un principio, tuvimos la idea de presentar una especie de cronología del progreso de la investigación ortográfica, y colocarla en este trabajo, sustituyendo, posiblemente, a los capítulos III, IV y V. Al final, decidimos manejar la presentación de los temas tal y como está aquí, pues de este modo, la cronología sólo sirve como una guía organizada, que no pretende desplazar el valor del resto de los capítulos. Por otra parte, para evitar que este escrito quedara atiborrado con múltiples referencias a los artículos que consultamos, o que consideramos de relevancia para aquellos lectores interesados en un estudio posterior del tema, preferimos concentrar en esta sección todas las referencias, permitiendo que la bibliografía sólo quede integrada por aquellos artículos con los que tuvimos un mayor acercamiento. En fin, comencemos pues con la cronología.

1959: W. W. Bledsoe e I. Browning desarrollan, para una aplicación de reconocimiento óptico, una metodología dividida en dos etapas: la primera correspondiente al reconocimiento individual de caracteres, y la segunda especializada en el reconocimiento de palabras completas. Según sus conclusiones, el trabajo sobre palabras completas perfeccionó el tratamiento de los caracteres individuales, pues, como era de esperarse, la utilización del diccionario concede una ventaja adicional sobre el simple reconocimiento óptico. (Publicación relacionada: *BLEDSOE, W. W. and BROWNING, I. 1959. Pattern recognition and reading by machine. In Proceedings of the Eastern Joint Computer Conference, Vol. 16, 255-232*).

1964: F. J. Damerau concluye que el ochenta por ciento del total de errores ortográficos posibles, incluye una, y sólo una, de las siguientes cuatro instancias de error: inserción, supresión, sustitución o transposición de letras. (Publicación relacionada: *DAMERAU, F. J. 1964. A technique for computer detection and correction of spelling errors. Commun. ACM 7, 3 March, 171-176*).

1964: F. J. Damerau implementa el primer algoritmo de corrección basado en la técnica de edición de mínima distancia. (Publicación relacionada: *DAMERAU, F. J. 1964. A technique for computer detection and correction of spelling errors. Commun. ACM 7, 3 March, 171-176*).

1966: V. I. Levenshtein diseña un algoritmo similar al que hizo Damerau en 1964, para la corrección de supresiones, inserciones y transposiciones simples. (Publicación relacionada: *LEVENSHTEIN V. I. 1966. Binary codes capable of correcting deletions, insertions and reversals. Sov. Phys. Dokl. 10, February, 707-710*).

1971: R. E. Gorin diseña un corrector ortográfico para una máquina *DEC-10*. En este trabajo, Gorin programa, por primera vez en la historia, una técnica "inversa" a la distancia mínima de edición, y constituye así la metodología que hasta la fecha sigue siendo la más común de las técnicas relacionadas con distancias mínimas. (Publicación relacionada: *GORIN R. E. 1971. SPELL: A spelling checking and correction program. Online documentation for the DEC-10 computer*).

1972: L. D. Harmon observa que el 42% de todos los bigramas del idioma inglés, nunca figuran en ninguna palabra, y que la sustitución aleatoria de una letra en una palabra, produce al menos un nuevo bigrama que, en el 70% de los casos, posee una probabilidad de existencia de cero. (Publicación relacionada: *HARMON L. D. 1972. Automatic recognition of print and script. Proc. IEEE 60, October, 1165-1176*).

1972: P. Tenczar y W. Golden diseñan el sistema de tutoría computacional *PLATO*, para el *Laboratorio de investigaciones de educación basada en computadoras de la Universidad de Illinois*. Este trabajo está hecho con ayuda de la técnica de claves similares. (Publicación relacionada: *TENCZAR, P. and GOLDEN, W. 1972. CERL Report X-35. Computer-Based Education Research Lab., Univ. of Illinois, Urbana, Ill*).

1973: T. K. Landauer y L. A. Streeter realizan un estudio tras el cual concluyen que las palabras de baja longitud, tienden a tener más errores ortográficos sencillos que las palabras de gran longitud. (Publicación relacionada: *LANDAUER, T. K. and STREETER, L. A. 1973. Structural differences between common and rare words. J. Verbal Learn. Verbal Behav. 12, 119-131*).

1973: G. D. Forney Jr. desarrolla, bajo el auspicio de la programación dinámica, el *algoritmo de Viterbi*, método que se considera como la más eficiente forma de combinar el cálculo de probabilidades de confusión, con el cálculo de probabilidades de transición. (Publicación relacionada: *FORNEY, G. D., Jr. 1973. The Viterbi algorithm. Proc. IEEE 61, 3, March, 268-278*).

1974: R. A. Wagner introduce, por primera vez, las nociones de programación dinámica a las técnicas de distancia de edición mínima. (Publicación relacionada: *WAGNER, R. A. 1974. Order-n correction for regular languages. Commun. ACM 17, 5, May, 265-268*).

1974: R. A. Wagner y M. J. Fischer generalizan el algoritmo de Levenshtein, de 1966, de modo que también pudiera corregir errores de calidad múltiple. (Publicación relacionada: *WAGNER R. A. FISCHER M. J. 1974. The string-to-string correction problem. J. ACM 21, 1, January, 168-178*).

1974: E. M. Riseman y A. R. Hanson demuestran que la mejor manera de capturar la sintaxis de un diccionario a través de n-gramas, es particionarlo en subconjuntos (subdiccionarios), procurando que todas las palabras de igual longitud queden contenidas en un único subconjunto, para luego construir un n-grama binario posicional por cada elemento de la partición. Bajo estos principios, Riseman y Hanson programaron un sistema que más tarde evaluarían contra un bloque de errores ortográficos pertenecientes a un total de 2'755 palabras de seis letras. Sus resultados marcaron que los arreglos trigramas posicionales, son capaces de detectar el 98.6% de los errores, y de corregir el 62.4% de los mismos. (Publicación relacionada: *RISEMAN, E. M. and HANSON, A. R. 1974. A contextual postprocessing system for error correction using binary n-grams. IEEE Trans. Comput. C-23, May, 480-493*).

1974: E. M. Riseman y A. R. Hanson en una continuación del estudio antes citado, aceptan que la desventaja de los correctores basados exclusivamente en n-gramas, es que sus sistemas de generación de candidatos a servir de corrección, pueden llegar a producir palabras inexistentes en el idioma, puesto que no las obtienen a través de un diccionario, sino a través de la simple inspección de sucesiones válidas de caracteres. (Publicación relacionada: *RISEMAN, E. M., and HANSON, A. R. 1974. A contextual postprocessing system for error correction using binary n-grams. IEEE Trans. Comput. C-23, May, 480-493).*

1975: A. V. Aho y M. J. Corasick diseñan un método de creación de autómatas de estado finito, los cuales representan pequeños conjuntos de términos que luego, con intereses de corrección ortográfica, comparan con archivos de texto. (Publicación relacionada: *AHO, A. V., and CORASICK, M. J. 1975. Fast pattern matching: An aid to bibliographic search. Commun. ACM 18, 6, June, 333-340*).

1975: R. Morris y L. L. Cherry construyen trigramas de frecuencia estadística especiales para cada documento que pretenden corregir, bajo el supuesto de que los errores ortográficos siempre contienen trigramas poco frecuentes, incluso para el documento mismo. (Publicación relacionada: *MORRIS, R., and CHERRY, L. L. 1975. Computer detection of typographical errors. IEEE Trans. Profess. Commun. PC-18, 1, 54-63*).

1976: A. R. Hanson, E. M. Riseman y E. Fischer descubren que los arreglos trigramas binarios posicionales, proporcionan una mayor exactitud en la corrección de textos producidos por reconocimiento óptico, que cualesquiera de los demás tipos de n-gramas. (Publicación relacionada: *HANSON, A. R., RISEMAN, E. M. and FISCHER, E. 1976. Context in word recognition. Patt. Recog. 8, 35-45*).

1977: F. E. Muth Jr. y A. L. Tharp inventan una heurística que permite almacenar, caracter por caracter, un diccionario completo dentro de un árbol pseudo-binario, con el objeto de hallar una alternativa que favoreciera los tiempos de búsqueda en diccionario, durante la aplicación de los algoritmos de la distancia mínima de edición. Según Muth y Tharp, su sistema basado en estos principios, corrigió el 97% de los errores de un texto de 1'487 palabras. (Publicación relacionada: *MUTH, F. E., Jr., and THARP, A. L. 1977. Correcting human error in alphanumeric terminal input. Inf. Process. Manage. 13, 329-337*).

1977: J. R. Ullman diseña una metodología que encuentra, para cada falta de ortografía, todas las palabras correctas que difieren de ella por a lo más dos inserciones, supresiones, sustituciones o inversiones. Esta técnica no sólo usó planteamientos algorítmicos, sino también innovaciones de hardware, como lo fue el procesamiento en paralelo de n-gramas binarios. (Publicación relacionada: *ULLMAN, J. R. 1977. A binary n-gram technique for automatic correction of substitution, deletion, insertion and reversal errors in words. Comput. J. 20, 141-147*).

1978: M. K. Odell y R. C. Russell programan el sistema *SOUNDEX*, como apoyo a un corrector fonético, y para ello se basan en la técnica de claves similares. (Publicación relacionada: *ODELL, M. K., and RUSSELL, R. C. 1978. U.S. Patent Numbers 1,261,167 (1918) and 1,435,663 (1992). U.S. Patent Office, Washington, D.C.*).

1979: R. Shingal y G. T. Toussaint asumen que la combinación de los métodos de búsqueda en diccionario, con la información probabilística, debe de generar excelentes resultados, siempre y cuando se aprovechen las ventajas de uno para suplir las fallas del otro, pues el trabajo con diccionarios tiene siempre porcentajes de error bajos, al lado de grandes demandas de memoria y complejidad computacional, mientras que los métodos de Markov tienen justo la característica inversa. (Publicación relacionada: *SHINGAL, R., and TOUSSAINT, G. T. 1979. A bottom-up and top-down approach to using context in text recognition. Int. J. Man-Machine Stud. 11, 201-212*).

1980: J. L. Peterson estudia la posible distribución, en diferentes niveles de memoria (memoria cache, memoria regular y memoria secundaria), de las palabras de los diccionarios empleados por las técnicas de corrección de ortográfica. (Publicación relacionada: *PETERSON, J. L. 1980. Computer programs for detecting and correcting spelling errors. Commun. ACM 23, 12, December, 676-684*).

1980: A. M. Wing y A. D. Baddeley trabajan con ensayos hechos a mano por estudiantes del *Colegio de Cambridge*, los cuales acumulaban un total de 80'000 palabras y 1'185 faltas de ortografía. Ellos descubrieron que, cuando menos, el 30% de los errores de los estudiantes sólo puede ser detectado por correctores dependientes de contexto. (Publicación relacionada: *WING , A .M. and BADDELEY, A. D. 1980. Spelling errors in handwriting: A corpus and distributional analysis. In cognitive processes in spelling, U. Frith, Ed. Academic Press, London*).

1980: G. Deloche y F. Debili demuestran que tanto en el francés como en el inglés, el 96% de los anagramas producidos por palabras propias de diccionarios de más de 20'000 vocablos, sin contar conjugaciones de verbos ni declinaciones morfológicas o inflexiones gramaticales, tienen soluciones únicas, esto es, que en el 96% de los casos, sólo las palabras originales son correctas, mientras que todos sus anagramas son errores ortográficos. (Publicación relacionada: *DELOCHE, G. and DEBILI, F. 1980. Order information redundancy of verbal codes in French and English: Neurolinguistic implications. J. Verbal Learn. Verbal Behav. 19, 525-530*).

1981: E. M. Zamora, A. Zamora y J.J. Pollock crean variaciones sobre las técnicas de corrección basadas en n-gramas, al decidir que éstos dejen de ser matrices binarias, para poder llenar sus entradas con frecuencias estadísticas o datos probabilísticos. (Publicación relacionada: *ZAMORA, E. M., ZAMORA, A. and POLLOCK, J.J. 1981. The use of trigram analysis for spelling error detection. Inf. Process. Manage. 17, 6, 305-316*).

1981: M. R. Dunlavey programa su innovador algoritmo llamado *SPROFF*, en el que guarda al diccionario como una inmensa máquina de estado finito, misma que reconoce a todas las palabras del diccionario, y solamente a ellas. Este diseño está basado en la integración de *tries* a los métodos computacionales de corrección. (Publicación relacionada: *DUNLAVEY, M. R. 1981. On spelling correction and beyond. Commun. ACM 24, 9, September, 608*).

1981: R. H. Boivie sigue una técnica semejante a la del algoritmo *SPROOF* de M. R. Dunlavey, y a partir de ella genera su sistema *da* de asistencia al directorio. (Publicación relacionada: *BOIVIE, R. H. 1981. Directory assistance revisited. AT&T Bell Labs. Tech. Mem. June 12, 1981*).

1981: W. D. Taylor utiliza las investigaciones hechas por Boivie en el mismo año, para realizar la programación de uno de los más exitosos sistemas basados en la distancia de edición mínima: el sistema *grope*. (Publicación relacionada: *TAYLOR, W. D. 1981. GROPE-A spelling error correction tool. AT&T Bell Labs. Tech. Mem.*).

1982: J. J. Hull y S. N. Srihari estiman que la representación de 5'000 vocablos de siete letras cada uno, en trigramas binarios posicionales, permite detectar el 98% de los errores de sustitución cometidos por los lectores ópticos en vocablos de ese tipo. Aprovechando esta observación, y otros conocimientos y perspectivas, Hull y Srihari aplican el *algoritmo de Viterbi* al problema la corrección ortográfica por computadora. (Publicación relacionada: *HULL, J. J. and SRIHARI, S. N. 1982. Experiments in text recognition with binary n-gram and Viterbi algorithms. IEEE Trans. Patt. Anal. Machine Intell. PAMI-4, 5, September, 520-530*).

1982: M. J. Hawley estudia la *métrica de Damerau-Levenshtein*, con la intención de crear un corrector para la interfase del lenguaje de comandos de *UNIX*. (Publicación relacionada: *HAWLEY, M. J. 1982. Interactive spelling correction in UNIX: The METRIC Library. AT&T Bell Labs. Tech. Mem., August 31*).

1983: Primera publicación del ***Webster's new world misspeller's dictionary***, un diccionario que en vez de contener palabras correctas, posee las más comunes faltas de ortografía de los hablantes de lengua inglesa.

1983: E. J. Yannakoudakis y D. Fawthrop analizan 1'377 errores ortográficos presentes en obras literarias, y hallan que la frecuencia de aparición de los errores en palabras cortas es muy escaso, tan sólo del 1.5% (Publicación relacionada: *YANNAKOUDAKIS, E. J., and FAWTHROP, D. 1983. An intelligent spelling corrector. Inf. Process. Manage. 19, 12, 101-108*).

1983: E. J. Yannakoudakis y D. Fawthrop concluyen que una significativa porción de las faltas de ortografía del idioma inglés, puede ser corregida por la simple aplicación de diecisiete reglas heurísticas preparadas por ellos. (Publicación relacionada: *YANNAKOUDAKIS, E. J., and FAWTHROP, D. 1983. The rules of spelling errors. Inf. Process. Manage. 19, 2, 87-99*).

1983: I. Durham, D. A. Lamb y J. B. Saxe diseñan una técnica "inversa" a la distancia mínima de edición, y la incluye en su corrector para un lenguaje de comandos. (Publicación relacionada: *DURHAM, I., LAMB, D. A., and SAXE J. B. 1983. Spelling correction in user interfaces. Commun. ACM 26, 10, October, 764-773*).

1983: *El Grupo de investigación mecanográfica LNR*, desarrolla un modelo de simulación mecanográfica por computadora, de donde sus investigadores observan que la mayoría de los errores de expertos mecanógrafos son inserciones, en los que suelen incurrir por golpear dos letras adyacentes del teclado al mismo tiempo, a causa de un exceso de velocidad en la transcripción. (Publicación relacionada: *GENTNER, D. R., GRUDIN, J., LAROCHELLE, S., NORMAN, D. A., and RUMELHART, D. E. 1983. Studies of typing from the LNR typing research group. In cognitive aspects of skilled typewriting, W. E. Cooper, Ed. Springer-Verlag, New York*).

1983: A. Zamora y J.J. Pollock trabajan sobre 50'000 palabras erradas extraídas de un total de veinticinco millones de vocablos pertenecientes a textos del *Chemical Abstracts Service*. Según su estudio, 23% del total de errores ortográficos se presenta en la tercera posición de una palabra, y, salvo contadas excepciones, una misma falta de ortografía nunca se repite, dos veces, en un mismo texto. (Publicación relacionada: *ZAMORA, A. and POLLOCK, J. J. 1983. Collection and characterization of spelling errors in scientific and scholarly text. J. Amer. Soc. Inf. Sci. 34, 1, 51-58*).

1983: R. C. Angell, G. E. Freund y P. Willett explican el desarrollo de una técnica de corrección ortográfica que calcula una medida de similitud, basada en una función que determina el número de trigramas binarios no posicionales comunes a una palabra errada y a una palabra del diccionario. A esta función se le nombra *coeficiente de Dice*, y su exactitud fue del orden del 76% sobre un total de 1'544 errores, disponiendo de un diccionario de 64'636 palabras. (Publicación relacionada: *ANGELL, R. C., FREUND, G. E. and WILLETT, P. 1983. Automatic spelling correction using a trigram similarity measure. Inf. Process. Manage. 19, 255-261*).

1983: D. J. Burr amplió la metodología desarrollada por Bledsoe y Browning en 1959, con la intención de diseñar un corrector para una aplicación de reconocimiento de texto escrito a mano. Dado que los requerimientos computacionales del sistema de Bledsoe y Browning, crecen a medida que crece el diccionario, y dado que Burr no podía prescindir de un enorme léxico, le adicionó a la técnica de Bledsoe y Browning una rutina de procesamiento morfológico. (Publicación relacionada: *BURR, D. J. 1983. Designing a handwriting reader. IEEE Trans. Patt. Anal. Machine Intell. PAMI-5, 5, September, 554-559*).

1983: S. N. Srihari, J. J. Hull y R. Choudhari programan un sistema de corrección basado en la teoría de probabilidades. En dicho sistema, el diccionario es guardado en un *trie*, y la información probabilística es manejada a través del *algoritmo de Viterbi*. Sus resultados indicaron que el *algoritmo de Viterbi*, usado sin el apoyo del diccionario, apenas es capaz de corregir el 35% de los errores en los que incurrió el lector óptico utilizado, mientras que el puro diccionario, sin la ayuda del algoritmo, corrigió eficientemente el 82% de los errores, en tanto que la utilización simultánea de ambas herramientas, alcanzó una exactitud del 87%. (Publicación relacionada: *SRIHARI, S. N., HULL, J. J., and CHOUDHARI, R. 1983. Integrating diverse knowledge sources in text recognition. ACM Trans. Office Inf. Syst. 1, 1, January, 68-87*).

1984: A. Zamora y J.J. Pollock encuentran que solamente un 6% de un total de 50'000 errores ortográficos de la lengua inglesa, pertenecientes al *Chemical Abstracts Service*, constituyen errores ortográficos de calidad múltiple. Luego, emplean esta y otras deducciones para idear una técnica de claves similares. Este proyecto concluyo con el desarrollo de una útil herramienta de corrección de errores sencillos, llamada *SPEEDCOP*, sistema que alcanza una efectividad del orden del 77% al 96% de precisión en la corrección de faltas sencillas. (Publicación

relacionada: *ZAMORA, A. and POLLOCK, J. J. 1984. Automatic spelling correction in scientific and scholarly text. Commun. ACM 27, 4, April, 358-368*).

1984: R. L. Kashyap y B. J. Oomen programan un sistema de corrección ortográfica, basados en el supuesto de que todos los errores de un texto pueden reducirse a una combinación de simples sustituciones, inserciones o supresiones. Su algoritmo fue recursivo, y permitía calcular la probabilidad de que dada una palabra ortográficamente errónea, esta fuese una versión deformada de alguna de las palabras del diccionario. (Publicación relacionada: *KASHYAP, R. L., and OOMEN, B. J. 1984. Spelling correction using probabilistic methods. Patt. Recog. Lett. 2, 3, March, 147-154*).

1986: J. L. Peterson determina que al menos el 16% de los errores simples, convierten a una palabra válida en otra nueva palabra válida, pero diferente de la original, manifestando que ese 16% pasará inadvertido para los detectores independientes de contexto. (Publicación relacionada: *PETERSON, J. L. 1986. A note on undetected typing errors. Commun. ACM 29, 7, July, 633-637*).

1986: D. E. Walker y R. A. Amsler comparan ocho millones de palabras del *New York Times* contra el *Merriam-Webster Seventh Collegiate Dictionary*. El resultado fue que el 61% de las palabras del diccionario no aparecieron en el texto, mientras que el 64% de las palabras del texto no aparecieron en el diccionario. (Publicación relacionada: *WALKER, D. E., and AMSLER, R. A. 1986. The use of machine-readable dictionaries in sublanguage analysis. In analysing language in Restricted Domains: Sublanguage Description and Processing. Lawrance Erlbaum, Hillsdale, N. J., 69-83*).

1986: R. Mitton prepara un diccionario especializado en labores de corrección ortográfica por computadora. Su diccionario contó con 38'000 infinitivos y 68'000 conjugaciones derivadas de ellos. (Publicación relacionada: *MITTON, R. 1986. A partial-dictionary of English in computer-usable form. Lit. Ling. Comput. 1, 4, 214-215*).

1986: D. E. Rumelhart, G. E. Hinton y R. J. Williams exponen la primera implementación del algoritmo de propagación hacia atrás (*Back propagation algorithm*), el cual es considerado, hoy en día, como el algoritmo más ampliamente utilizado en el entrenamiento de redes neuronales. (Publicación relacionada: *RUMELHART, D. E., HINTON, G. E. and WILLIAMS, R. J. 1986. Learning internal representations by error propagation. In parallel distributed processing: Explorations in the microstructure of cognition, Ed. Bradford Books/MIT Press*).

1987: J. Henseler, J. C. Scholtes y C. R. J. Verdoest elaboran una técnica de procesamiento en paralelo para el reconocimiento óptico y la corrección ortográfica. Su metodología requirió de una máquina *16-processor hypercube*, y en un examen hecho sobre una pequeña muestra de texto, obtuvo una exactitud del 95% en el reconocimiento inicial de caracteres, y del 100% en el reconocimiento posterior a la corrección. (Publicación relacionada: *HENSELER, J., SCHOLTES, J. C., and VERDOEST, C. R. J. 1987. The design of a parallel knowledge-based optical character recognition system. Master of Science Theses. Dept. of Mathematics and Informatics, Delft Univ. of Technology*).

1987: M. A. Bickel combina la utilización de monogramas con medidas léxicas para métodos de distancia vectorial, y diseña un sistema administrativo en el cual se manejaban los nombres propios de los empleados adscritos a una cierta dependencia pública. Su diccionario consistió de aproximadamente mil nombres propios, y se pretendía que la computadora siempre devolviera el nombre que el usuario quiso teclear, aunque lo hubiese tecleado mal. En numerosas pruebas que le fueron practicadas, el sistema de Bickel nunca descendió del 95% de efectividad en la corrección. (Publicación relacionada: *BICKEL, M. A. 1987. Automatic correction to misspelled names: A fourth-generation language approach. Commun. ACM 30, 3, March, 224-228*).

1987: S. Kahan, T. Pavlidis y H. S. Baird integran el cálculo de probabilidades de confusión a los métodos de búsqueda en diccionario, para mejorar la calidad del reconocimiento óptico. En sus conclusiones, reportaron que un sistema programado bajo estas especificaciones, obtiene porcentajes de exactitud del orden del

97%, aunque no tomaron en cuenta aquellas parejas de caracteres indistinguibles para un dispositivo lector, como por ejemplo, *O* y *0* (la letra *O* y el cero). (Publicación relacionada: *KAHAN, S., PAVLIDIS, T. AND BAIRD, H. S. 1987. On the recognition of characters of any font size. IEEE Trans Patt. Anal. Machine Intell. PAMI-9, 9, 274-287*).

1987: R. Mitton estudia 170'016 palabras contenidas en textos escritos a mano por estudiantes de preparatoria, y descubre que apenas el 3% de los errores son de calidad múltiple. Además aseguró que el 40% de los errores existentes, sólo podrían haber sido detectados por correctores dependientes de contexto. (Publicación relacionada: *MITTON, R. 1987. Spelling checkers, spelling correctors, and the misspellings of poor spellers. Inf. Process. Manage. 23, 5,, 495-505*).

1988: K. Kukich aplica las redes neuronales en las aplicaciones de síntesis de texto a voz, utilizando la arquitectura normal de una red de *propagación hacia atrás*. El sistema que produce bajo esta condiciones, alcanza una exactitud del orden del 75%. (Publicación relacionada: *KUKICH, K. 1988. Back-propagation topologies for sequence generation. In Proceedings of the IEEE International Conference on Neural Networks, vol. 1, San Diego, California, July, 24-27*).

1988: R. J. Elliot define clases de equivalencia de afijos, basadas en semejanzas ortográficas de las palabras, con la intención de implementar un avanzado sistema de procesamiento morfológico. (Publicación relacionada: *ELLIOT, R. J. 1988. Annotating spelling list words with affixation classes. AT&T Bell Labs. Int. Mem. Dec. 14*).

1988: L. G. Means produce un corrector ortográfico basado en reglas, para un sistema de entrada de datos en lenguaje natural, dentro del cual se incluía una amplia incidencia de abreviaturas y expresiones en jerga especializada. (Publicación relacionada: *MEANS, L. G. 1988. On your computer read this. In Proceedings of the 2nd Applied Natural Language Processing Conference, Austin, Texas, February, ACL, 93-100*).

1988: A. Goshtasby y R. W. Ehrich crean la técnica de la relajación (*relaxation technique*), con el objeto de combinar, eficientemente, las probabilidades de confusión con las de transición, en el ambiente de la corrección ortográfica. La complejidad computacional de esta técnica, por cada palabra, fue estimada por sus creadores en $10n^2$, donde n es el número de letras de la palabra. (Publicación relacionada: *GOSHTASBY, A. and EHRICH, R. W. 1988. Contextual word recognition using probabilistic relaxation labeling. Patt. Recog. 21, 5, 455-462*).

1988: B. Van Berkel y K. DeSmedt inventan la técnica del análisis trifono (*triphone analysis*), para una aplicación que involucraba el empleo de nombres propios, los cuales, en la mayoría de los casos, se convertían en faltas de ortografía cuando eran escritos con ayuda de la intuición fonética. Este sistema corrigió el 94% de los errores existentes en 123 apellidos alemanes mal escritos, y dispuso de un diccionario formado por 254 apellidos alemanes aleatoriamente tomados del directorio telefónico. (Publicación relacionada: *VAN BERKEL, B. and DESMEDT, K. 1988. Triphone analysis: A combined method for the correction of orthographical and typographical errors. In Proceedings of the 2nd Applied Natural Language Processing Conference, Austin, Texas, February, Association for Computational Linguistics, ACL*).

1988: T. Oshika, F. Machi, B. Evans y J. Tom utilizan la información probabilística como preprocesador de correcciones, a través de la implementación de lo que llamaron Modelo de Markov Oculto (*Hidden Markov model*). (Publicación relacionada: *OSHIKA, T., MACHI, F., EVANS, B. and TOM, J. 1988. Computational techniques for improved name search. In Proceedings of the 2nd Annual Aplied Natural Language Conference, Austin, Texas, February. ACL. 203-210*).

1988: R.M.K. Sinha y B. Prasada aceptan que es imposible garantizar que un diccionario sea, siempre, lo suficientemente completo como para contener todas las palabras que se incluyen en un texto. Bajo este supuesto, programan un sistema que mezcla las técnicas probabilísticas con los métodos de búsqueda en diccionario, y, además, una buena cantidad de heurísticas. En un examen hecho sobre 5'000 palabras de texto generado a través

de máquinas de escribir e impresoras, con diferentes tipos y estilos de letras, el sistema de Sinha y Prasada obtuvo una eficiencia superior al 98%, para el reconocimiento individual de caracteres, y de alrededor del 97% para palabras completas, sin contar signos de puntuación. (Publicación relacionada: *SINHA, R. M. K. and PRASADA, B. 1988. Visual text recognition through contextual processing. Patt. Recog. 21, 5, 463-479*).

1989: G. Sampson pone a prueba el diccionario hecho por Mitton en 1986, al pretender corregir con él un texto de 50'000 palabras. Sampson determinó que el trabajo de Mitton fue excelente, pues solamente 3.24% de las palabras del texto no aparecieron en su diccionario. (Publicación relacionada: *SAMPSON, G. 1989. How fully does a machine-usabledictionary cover English text. Lit. Ling. Comput. 4, 1, 29-35.*).

1989: F. J. Damerau y E. Mays descubren que el incremento de palabras en un diccionario computarizado, capacita a los sistemas para detectar errores no percibidos por diccionarios pequeños. (Publicación relacionada: *DAMERAU, F. J. and MAYS, E. 1989. An examination of undetected typing errors. Inf. Process. Manage. 25, 6, 659-664*).

1989: V. Cherkassky y N. Vassilas aplican la metodología de redes neuronales al problema de la corrección ortográfica de la escritura de nombres propios. La experiencia extraída de este trabajo, les permite afirmar que el número óptimo de unidades ocultas de las que debe de disponer una red, para aplicaciones de automatización ortográfica, debe de ser muy cercano al número total de palabras que habrá en el diccionario. (Publicación relacionada: *CHERKASSKY, V. and VASSILAS, N. 1989. Back propagation networks for spelling correction. Neural Net. 1, 3, July, 166-173*).

1990: Y. C. Tsao analiza 130'000 palabras tecleadas por usuarios del *Servicio de telecomunicaciones para sordos*, y calcula que el porcentaje de errores reconocido por un detector independiente, es de tan solo el 5%. (Publicación relacionada: *TSAO, Y. C. 1990. A lexical study of sentences typed by hearing-impaired TDD users. In Proceedings of the 13th International Symposium on Human Factors in Telecommunications, Turin, Italy, September, 197-201*).

1990: M. D. Kernighan, K. W. Church y W. A. Gale diseñan un sistema de corrección de errores ortográficos sencillos, al que posteriormente llamaron *correct*. Para ello, crearon una base de datos de errores ortográficos genuinos, extraídos de palabras generadas por la agencia *Associated Press*, y luego compilaron cuatro diferentes matrices de confusión, una por cada uno de los cuatro tipos de errores sencillos. También estimaron las probabilidades de confusión que sus múltiples estadísticas les permitieron determinar, y, al final, usaron una técnica "inversa" a la distancia de edición mínima, para obtener un conjunto de posibles correcciones, al cual le aplicaron métodos de cálculo bayesiano con el objeto de ordenar a los candidatos. (Publicación de errores: *KERNIGHAN, M. D., CHURCH, K. W., and GALE, W. A. 1990. A spelling correction program based on a noisy channel model. In Proceedings of COLING-90, the 13th International Conference on Computational Linguistics, vol. 2, Helsinki, Finland, Hans Karlgren, Ed. 205-210*).

1990: K. Kukich lleva a cabo un estudio comparado sobre el tema de la corrección ortográfica computarizada, de manera que sea posible verificar cuál de los diferentes métodos de corrección posee una mayor exactitud. (Publicación relacionada: *KUKICH, K. 1990. A comparision of some novel and traditional lexical distance metrics for spelling correction. In Proceedings of INNC-90-Paris (paris, France, July), 309-313*).

1990: V. Cherkassky y P. Dahl prueban que los errores por transposición son más eficientemente resueltos por los monogramas, sin importar la longitud de las palabras en las que se hallen, mientras que los bigramas y los trigramas, permiten obtener mejores porcentajes de eficiencia en la corrección ortográfica de los otros tres tipos de errores sencillos. (Publicación relacionada: *CHERKASSKY, V., and DAHL, P. 1990. Combined encoding in associative spelling checkers. Univ. of Minnesota EE Dept. Tech. Rep.*).

1990: S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer y R. Harshman emplean la técnica *SVD*, pero con la particularidad de representar a las palabras del documento por un tipo especial de matrices,

llamadas matrices *término-por-documento*. El sistema que produjeron fue nombrado sistema de indexación semántica latente (*Latent semantic indexing*), debido a que la metodología *SVD* fue aprovechada para manifestar las relaciones semánticas inherentes entre los términos y los documentos. (Publicación relacionada: *DEERWESTER, S., DUMAIS, S. T., FURNAS, G. W., LANDAUER, T. K. and HARSHMAN, R. 1990. Indexing by latent semantic analysis. JASIS 41, 6, 391-407*).

1990: M. Gersho y R. Reiter preparan un sistema ortográfico orientado hacia la corrección de nombres de calles. En este trabajo, los investigadores combinaron el uso de redes de auto organización de tipo Kohonen (*Self-organizing Kohonen-type network*), con muchas pequeñas redes de propagación hacia atrás. Este sistema de Gersho y Reiter alcanzó una exactitud oscilante entre el 74% y 97%. (Publicación relacionada: *GERSHO, M. and REITER, R. 1990. Information retrieval using self-organizing and heteroassociative supervised neural networks. In Proceedings of IJCNN, San Diego, California, June*).

1991: M. A. Jones, G. A. Story y B. W. Ballard determinan que los errores producidos por los reconocedores ópticos, varían ampliamente de un reconocedor a otro, y de una calidad de texto a otra, además de que los errores de los dispositivos ópticos no son errores uno a uno. Posteriormente, con base en todo esto, programan un postprocesador especializado en reconocimiento óptico, en el cual integran un tratamiento bayesiano para un conjunto de predicciones hechas a partir de una multitud de fuentes de conocimiento léxico. (Publicación relacionada: *JONES, M. A., STORY, G. A. and BALLARD, B. W. 1991. Integrating multiple knowledge sources in a Bayesian OCR post-processor. In Proceedings of IDCAR-91, St. Malo, France, 925-933*).

1991: M. D. Kernighan utiliza cuarenta y cuatro millones de palabras producidas por la *Associated Press*, con la intención de crear una técnica que le permita hallar todos los posibles vocablos del diccionario formados por una, y sólo una, inserción, supresión, sustitución o transposición de las letras de palabras mal escritas. Posteriormente, Kernighan prepara un corrector ortográfico con alta ingerencia de la teoría de probabilidades, y después le adiciona una técnica "inversa" a la distancia mínima de edición para generar sus candidatos a servir de corrección. (Publicación relacionada: *KERNIGHAN, M. D. 1991. Specialized spelling correction for a TDD system. AT&T Bell Labs. Tech. Mem., August, 30*).

1992: V. Cherkassky, N. Vassilas y K. Fassett utilizan un teorema del álgebra lineal para probar que una matriz inversa generalizada, se satura cuando el número de entradas del diccionario se aproxima a la dimensión del espacio léxico, razón por la que concluyen que el modelo *CMM* es más eficiente para la corrección ortográfica que el método de la *inversa generalizada*. (Publicación relacionada: *CHERKASSKY, V., VASSILAS, N., and Fassett, K. 1991. Linear algebra approach to neural associative memories and noise performance of neural classifiers. IEEE Trans. Cmput. 40, 12, 1429-1435*).

1991: K. W. Church y W. A. Gale encuentran que el porcentaje de errores aparecidos en el texto noticioso de la agencia *Associated Press*, es del 0.05%, generando, semanalmente, un millón de palabras y 500 typos. (Publicación relacionada: *CHURCH, K. W. and GALE, W. A. 1991. Enhanced Good-Turing and cat-cal: Two new methods for estimating probabilities of English bigrams. Comput. Speech Lang.*).

1991: K. W. Church y W. A. Gale utilizan la "inversa" de la técnica de la distancia mínima de edición, para generar candidatos a corrección en un sistema diseñado con una metodología probabilística. (Publicación relacionada: *CHURCH, K. W. and GALE, W. A. 1991. Probability scoring for spelling correction. Stat. Comput. 1, 93-103*).

1991: A. K. Bocast vincula la técnica de la distancia mínima de edición con la de las claves similares, y de esta composición desprende el conocido algoritmo de la reconstrucción atómica (*token reconstruction*). (Publicación relacionada: *BOCAST, A. K. 1991. Method and apparatus for reconstructing a token from a Token Fragment. U. S. Patent Number 5,008,818, Design Services Group. Inc. McLean, Va.*).

1992: V. Cherkassky, N. Vassilas, G. L. Brodt y H. Wechsler practican estudios detallados en los que determinan que el uso de trigramas, en apoyo de los vectores de características léxicas, proporciona excelentes porcentajes de eficiencia (superiores al 90%), en los modelos de corrección basados en las técnicas *CMM*. (Publicación relacionada: *CHERKASSKY, V., VASSILAS, N., BRODT, G. L., and WECHSLER, H. 1992. Conventional and associative memory approaches to automatic spelling checking. Eng. Appl. Artif. Intell. 5, 3*).

1992: O. Matan, C. J. C. Burges, Y. LeCun y S. S. Denker aplican técnicas de redes neuronales en el problema del reconocimiento óptico de caracteres escritos a mano, con la intención de satisfacer demandas existentes en los sistemas de lectura de códigos postales, tarjetas de crédito y cheques bancarios. (Publicación relacionada: *MATAN, O.; BURGES, C. J. C., LECUN, Y. and DENKER, S. S. Multi-digit recognition using a space displacement neural network. In Advances in Neural Information Processing Systems, vol. 4, Ed. Morgan Kauffmann, San Mateo, California, 488-495*).

1992: K. Kukich revisa 40'000 palabras tecleadas por usuarios del *Servicio de telecomunicaciones para sordos*, y descubre en ellos un porcentaje de error del 6%. (Publicación relacionada: *KUKICH, K. 1992. Spelling correction for the telecommunications network for the deaf. Commun. ACM 35, 5, May, 80-90*).

APÉNDICE 2
ANÁLISIS DE N-GRAMAS

El primer paso en este análisis sobre los n-gramas de la lengua española, fue determinar las frecuencias de aparición que poseen los 33 monogramas, los 1'089 bigramas y los 35'937 diferentes trigramas que se pueden obtener a partir del alfabeto español. Para tal efecto, utilizamos el texto que compone estas mismas páginas, así como el de algunas cuantas hojas de un diccionario común y corriente de la lengua española. En cierto sentido, la determinación de los n-gramas del español que nosotros hicimos, se debe considerar como un indiscutible avance en la investigación ortográfica en México, puesto que no existen trabajos similares que hayan sido publicados a nivel académico. Tenemos que aceptar que no procesamos una cantidad estadísticamente significativa de n-gramas, ya que, por ejemplo, en cuanto a los bigramas, apenas trabajamos con 225'799 de ellos, lo cual no es mucho, sin embargo, es un trabajo que nos concede hablar con conocimiento de causa, al menos en un nivel elemental. Como ya dijimos, el conteo de n-gramas se realizó sobre el texto escrito de esta presentación, el cual, entre sus características particulares cuenta con las siguientes,

Número de Caracteres :	277'120
Número de Palabras Escritas :	52'905
Número de Líneas Escritas :	3'536
Número de Enunciados Escritos :	1'916
Número de Párrafos Escritos :	437
Longitud Promedio por Palabra Escrita :	5
Número Promedio de Palabras por Enunciado :	27
Máximo Número de Palabras por Enunciado :	129

Nuestro estudio estuvo limitado a los arreglos n-gramas no posicionales, con n variando entre uno y tres, pues ya si en un futuro decidimos trabajar con n-gramas asociados con índices de peculiaridad, o incluso con n-gramas posicionales, necesitaremos primero disponer de las frecuencias de aparición de los n-gramas, que es justamente lo que se obtiene con los arreglos que pretendimos usar. Con la intención de realizar un trabajo tan

detallado como se pudiera, decidimos asignarle un lugar especial a cada letra del alfabeto, y otro especial a cada una de sus posibles variantes ortográficas, esto es, por ejemplo, a la letra "*a*" se le concedió un lugar especial, y a la misma letra "*a*" pero acentuada, se le asignó otro lugar diferente. De esta forma, existen treinta y tres distintos caracteres en el alfabeto castellano, a saber,

ABCDEFGHIJKLMNÑOPQRSTUVWXYZÁÉÍÓÚÜ.

El primer resultado que devolvió nuestra investigación sobre los monogramas, fue bastante aceptable, en el sentido de que nos sirvió para comprobar una idea ya de antemano aceptada: la letra *E* es la letra más común de la lengua española, mientras que la letra *K* es la que menos apariciones tiene. Era de esperarse que las vocales superaran considerablemente a las consonantes, en cuanto al número de apariciones, dado que toda palabra del castellano posee por lo menos una vocal. No obstante, consonantes como la *N*, la *R* y la *S*, resultaron ser más frecuentes que la *I* y que la *U*, que son las vocales con menor número de apariciones registradas. Letras como la *W* y la *Ñ*, son de muy escasa presencia en las palabras comunes del español, razón por la cual podíamos esperar que ocuparan los últimos lugares de una tabla ordenada por frecuencia. He aquí nuestra tabla de monogramas, obsérvese que, para cada letra, agregamos su porcentaje de apariciones después del caracter "↔", y, en algunos casos, seguido del mismo caracter, su número exacto de apariciones.

Frecuencia de Monogramas de Texto Común					
K↔0.00%	W↔0.01%	Ü↔0.01%	Ñ↔0.09%	ú↔0.15%	Z↔0.26%
X↔0.31%	é↔0.33%	J↔0.36%	H↔0.42%	í↔0.44%	á↔0.53%
Y↔0.58%	V↔0.75%	F↔0.80%	ó↔0.84%	Q↔0.91%	G↔1.08%
B↔1.31%	M↔2.65%	P↔2.81%	U↔3.69%	T↔4.85%	L↔5.02%
D↔5.16%	C↔5.44%	I↔6.53%	N↔7.03%	R↔7.24%	S↔7.54%
	O↔8.49%	A↔11.38%	E↔12.99%		

Por lo que a los bigramas se refiere, encontramos que el bigrama *DE* es el más común de la lengua española, lo cual no nos extraña en lo más mínimo, toda vez que sabemos que la letra más común del idioma español es la letra *E*, mientras que la letra *D* es una de las seis consonantes con mayor número de apariciones. Más resultados sobre nuestra investigación de n-gramas, pueden ser consultados en las láminas especiales, elaboradas con gráficos para facilitar su comprensión, que aparecen al final de esta apéndice. Las tablas que esbozamos a lo largo de estas páginas, también ilustran muy bien los resultados que obtuvimos en nuestro trabajo, pero, desafortunadamente, para el caso de los bigramas y de los trigramas, no es posible mostrarlas en su totalidad, pues consumirían demasiado espacio. Luego entonces, la siguiente tabla sólo clasifica a los más frecuentes bigramas del idioma español, es decir, a aquellos con más de mil setecientas apariciones en los textos seleccionados. Note que cada bigrma va seguido de su número de apariciones y luego de su porcentaje de apariciones.

Frecuencia de Bigramas de Texto Común			
Ió↔1780↔0.79%	óN↔1795↔0.79%	DI↔1801↔0.80%	IN↔1803↔0.80%
TI↔1808↔0.80%	AN↔1867↔0.83%	SE↔1878↔0.83%	PA↔1942↔0.86%
PO↔1951↔0.86%	SI↔1951↔0.86%	EC↔1979↔0.88%	LO↔2023↔0.90%
IO↔2057↔0.91%	DA↔2069↔0.92%	DO↔2092↔0.93%	RO↔2305↔1.02%
NA↔2334↔1.03%	UN↔2344↔1.04%	ST↔2353↔1.04%	AD↔2420↔1.07%
TA↔2472↔1.09%	QU↔2528↔1.12%	CA↔2529↔1.12%	EL↔2540↔1.12%
IC↔2704↔1.20%	AL↔2800↔1.24%	TO↔2932↔1.30%	NT↔3245↔1.44%
UE↔3293↔1.46%	TE↔3305↔1.46%	ER↔3493↔1.55%	AR↔3508↔1.55%
ON↔3515↔1.56%	CO↔3679↔1.63%	CI↔3851↔1.71%	RE↔3939↔1.74%
OR↔4046↔1.79%	RA↔4075↔1.80%	AS↔4106↔1.82%	OS↔4251↔1.88%
LA↔4482↔1.98%	ES↔5947↔2.63%	EN↔6139↔2.72%	DE↔7474↔3.31%

Obsérvese que, en su mayoría, los bigramas más frecuentes incluyen siempre una vocal, lo cual nos confirma la conocida afirmación de que las vocales son más comunes que el resto de las letras del alfabeto. Por otra parte, si hubiésemos dispuesto de más textos que procesar, las apariciones de cada bigrama habrían aumentado, con toda seguridad, pero nos inclinamos a pensar que la distribución de los bigramas en la tabla mostrada, seguiría siendo la misma, o por lo menos muy parecida, en el sentido de que *DE* seguiría siendo el más frecuente de los bigramas, mientras que *LA*, *ES* y *EN*, seguirían estando muy cerca de él.

Un dato que puede parecerle insólito a las personas no conocedoras del tema de la corrección ortográfica, es que en la lengua española, en particular en nuestro trabajo, descubrimos doscientos cuarenta y siete bigramas que no aparecieron ni tan siquiera una vez. Entre esos bigramas de nula presencia en el idioma español se cuentan: *BB, ÉY, FY, HÑ, KJ, ÑG, PÜ, QW, TY, VJ, WE, XK, YÑ* y *ZL*. Otro dato igual de curioso, es que el número total de bigramas con menos de cien apariciones en el idioma español, es de ¡ochocientos setenta!, lo cual significa que el número de bigramas distintos que verdaderamente utilizamos durante la escritura de nuestra lengua, es bastante menor de lo que pensábamos. Si en total existen 1'089 bigramas, y 870 de ellos casi nunca se usan, entonces, en realidad, estamos hablando de que cualquier escritor modesto triunfa en nuestro idioma con tan sólo 219 combinaciones de parejas de letras.

De la misma forma en que estudiamos los monogramas y los bigramas de la lengua española, a través de estos párrafos, también estudiamos los trigramas de los mismos textos. El número total de trigramas existentes en el abecedario español es 35'937, de los cuales, para fines de un conveniente manejo computacional en una *P.C.*, descartamos a todos aquellos trigramas que incluyen vocales acentuadas. Así pues, analizamos solamente 19'683 trigramas, y descubrimos que apenas 1'929 de ellos tuvieron al menos una aparición. El total de trigramas procesados fue de 160'498, y, a continuación, mostramos una tabla compuesta por aquellos con más de seiscientas apariciones.

Frecuencia de Trigramas de Texto Común			
STE↔600↔0.37%	ORT↔602↔0.38%	ORR↔633↔0.39%	DEL↔636↔0.40%
NTO↔651↔0.41%	NCI↔652↔0.41%	IDA↔654↔0.41%	RRE↔654↔0.41%
COR↔659↔0.41%	STA↔666↔0.41%	PAL↔668↔0.42%	ALA↔675↔0.42%
OGR↔683↔0.43%	LAB↔685↔0.43%	ABR↔689↔0.43%	BRA↔691↔0.43%
RAS↔702↔0.44%	FIC↔708↔0.44%	ONA↔717↔0.45%	COM↔737↔0.46%
PRO↔820↔0.51%	ADA↔837↔0.52%	ARA↔846↔0.53%	ADO↔854↔0.53%
UNA↔857↔0.53%	CCI↔865↔0.54%	MEN↔891↔0.56%	IEN↔926↔0.58%
TRA↔934↔0.58%	REC↔950↔0.59%	PAR↔959↔0.60%	LAS↔1026↔0.64%
CIO↔1039↔0.65%	ACI↔1051↔0.65%	POR↔1052↔0.66%	ION↔1084↔0.68%
LOS↔1111↔0.69%	RES↔1248↔0.78%	EST↔1259↔0.78%	ICA↔1284↔0.80%
NTE↔1492↔0.93%	CON↔1557↔0.97%	ENT↔2232↔1.39%	QUE↔2252↔1.40%

Una de las afirmaciones que más nos esforzamos por resaltar, a lo largo de la exposición de los capítulos relativos a las técnicas de corrección, es aquella que establece lo ocioso que resulta hacer comparaciones entre los diferentes sistemas ortográficos, pues la eficacia de cada uno de ellos depende de una serie de parámetros que, por lo regular, no cumplen todas las metodologías a la vez, y por esa razón, no es posible determinar un terreno apropiado para las comparaciones. Uno de esos parámetros que cada técnica posee a su modo, es el del dominio del lenguaje sobre el cual trabaja. Así pues, la verdadera efectividad de una metodología, sólo puede ser juzgada cuando se aplica al espacio léxico específico para el cual fue creada. En este sentido, nosotros quisimos verificar la variación de la distribución de los n-gramas en razón del cambio del dominio léxico que los genera. Luego, una vez que obtuvimos las tres tablas hasta ahora mostradas, pensamos en analizar qué tan diferentes serían de otras que se obtuvieran, por ejemplo, a partir de un archivo en el que ninguna palabra estuviera repetida. La finalidad de los n-gramas, como de cualquier otra técnica de corrección de propósito general, es corregir textos reales, escritos por usuarios reales, los cuales manejan léxicos a veces muy grandes, o a veces escasos, pero que de

cualquier modo siempre repiten palabras durante su escritura. Por lo tanto, las apariciones de los n-gramas se cuentan a partir de textos reales, y la elevada frecuencia de uno de ellos se determina a partir de su elevada presencia en un texto real, quizá tan sólo porque forma parte de la más frecuente de las palabras de los escritos comunes, pero nunca porque así lo haya determinado un estudio del diccionario. A pesar de todo, nuestro interés era analizar la diferencia de una tabla obtenida a través de palabras repetidas, y de otra que se produjo sin la repetición de ningún vocablo. El archivo que generó las tres tablas anteriores, fue el compuesto por los capítulos que conforman este mismo trabajo, mientras que el archivo que produjo a las otras tres siguientes, fue hecho a partir de estos mismos capítulos, pero con la salvedad de que fueron sometidos a un proceso en el que cada repetición de una misma palabra es anulada, conservando, únicamente, una aparición por vocablo. Nuestra tabla de monogramas se muestra en seguida,

Frecuencia de Monogramas de Palabras Sin Repetición					
K↔0.00%	W↔0.01%	ü↔0.02%	ú↔0.10%	Ñ↔0.12%	Y↔0.16%
é↔0.24%	Q↔0.28%	X↔0.38%	J↔0.42%	H↔0.42%	á↔0.45%
Z↔0.47%	í↔0.63%	ó↔0.91%	F↔0.99%	G↔1.19%	V↔1.34%
B↔1.37%	P↔2.80%	U↔2.84%	M↔3.18%	L↔3.45%	D↔4.55%
T↔5.38%	C↔6.13%	S↔6.89%	N↔7.59%	O↔7.85%	R↔8.02%
	I↔8.44%	E↔11.32%	A↔12.06%		

Obsérvese que a pesar de que los dominios léxicos variaron, los dos últimos renglones de las dos tablas de monogramas que hicimos, cuentan casi con las mismas letras, salvo por estar dispuestas en otro orden. De hecho, si dejamos de lado el orden, los dos últimos renglones sólo difieren por una letra, lo cual significa que las letras más comunes en la escritura de este trabajo, seguirán siendo también las más comunes aún cuando se cancelen todas las repeticiones de cada palabra que fue escrita en este trabajo.

Veamos ahora la segunda tabla sobre bigramas.

Frecuencia de los Bigramas de Palabras Sin Repetición			
SI↔274↔0.69%	NC↔275↔0.69%	IE↔280↔0.71%	LA↔292↔0.74%
AM↔296↔0.75%	SE↔299↔0.76%	IO↔306↔0.77%	NA↔306↔0.77%
TO↔306↔0.77%	ID↔324↔0.82%	AC↔335↔0.85%	ST↔335↔0.85%
LE↔339↔0.86%	RI↔339↔0.86%	ME↔351↔0.89%	DI↔352↔0.89%
PR↔370↔0.93%	OR↔377↔0.95%	RO↔378↔0.95%	EC↔399↔1.01%
AL↔400↔1.01%	TI↔422↔1.07%	IC↔447↔1.13%	DE↔456↔1.15%
IN↔491↔1.24%	DA↔493↔1.25%	CA↔507↔1.28%	DO↔525↔1.33%
AS↔540↔1.36%	TA↔543↔1.37%	AN↔555↔1.40%	RA↔600↔1.52%
CO↔619↔1.56%	AD↔668↔1.69%	CI↔682↔1.72%	TE↔683↔1.73%
ON↔686↔1.73%	ER↔717↔1.81%	OS↔737↔1.86%	RE↔775↔1.96%
NT↔798↔2.02%	AR↔856↔2.16%	ES↔862↔2.18%	EN↔1075↔2.72%

Tres de los ocho bigramas más frecuentes en la escritura de las palabras no repetidas, son también tres de los ocho bigramas más frecuentes del texto común. Evidentemente, el bigrama *DE* aparece en alguna o algunas de las palabras que más se repiten en el texto, y es por eso que cuando se considera a todo el texto en su totalidad, sin quitarle nada, su frecuencia de aparición crece considerablemente, mientras que cuando se extraen las repeticiones de los vocablos, su frecuencia se reduce hasta ocupar el lugar veintiuno. Puede observarse que en las tablas de bigramas, la semejanza ya no fue tan sorprendente como en la de los monogramas, pero a pesar de eso, sigue existiendo. Vale la pena notar que son muy contadas las sucesiones de letras que solamente aparecen en una de las tablas, por lo que diera la impresión de sólo estar viendo dos distintas organizaciones de un mismo conjunto.

De algún modo, es como si nuestro idioma contará con una serie de sílabas de gran presencia fonética y escrita, las cuales no dependen del dominio del lenguaje sobre el que se esté trabajando, y por ello es que nuestros dos análisis están condenados a respetar un cierto parecido.

Las dos tablas que más difieren entre sí, son las correspondientes a los trigramas, lo cual indica que la caracterización del espacio léxico sobre el que se está trabajando, mejora en razón del aumento la longitud de los n-gramas. Esta es una deducción bastante lógica, puesto que es obvio que en la medida en que hacemos crecer la longitud de los n-gramas, nos acercamos más y más al tamaño real de las palabras, y si bien un texto no puede ser reconocido solamente por las letras que lo constituyen, sí puede ser caracterizado por las palabras que lo forman. Así pues, mientras que los monogramas, y hasta cierto punto los bigramas, no fueron suficientes para reconocer ninguna diferencia significativa entre los dos conjuntos de texto estudiados, los trigramas sí nos permitieron distinguir que los vocablos que produjeron a la primera tabla, tuvo que ser distinto del que sirvió como fuente para la segunda.

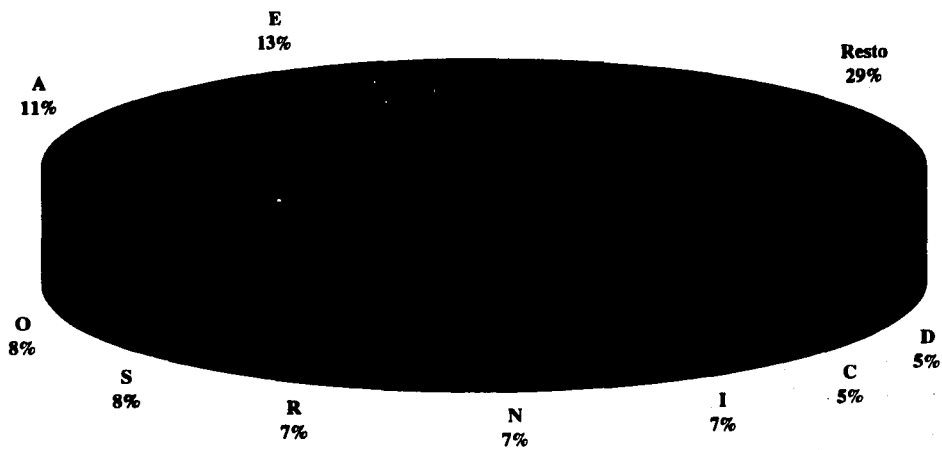
Frecuencia de los Trigramas de Palabras Sin Repetición

IZA↔97↔0.30%	NTO↔97↔0.30%	ICI↔98↔0.31%	INA↔98↔0.31%
ONA↔99↔0.31%	IDO↔101↔0.32%	NES↔102↔0.32%	FIC↔103↔0.32%
PAR↔103↔0.32%	STA↔104↔0.33%	TAR↔106↔0.33%	END↔110↔0.34%
TER↔113↔0.35%	DES↔115↔0.36%	CIA↔121↔0.38%	ENC↔121↔0.38%
ONE↔125↔0.39%	RON↔126↔0.39%	TEN↔128↔0.40%	TRA↔131↔0.41%
COM↔132↔0.41%	EST↔133↔0.42%	DAS↔136↔0.43%	ERA↔136↔0.43%
NCI↔137↔0.43%	IDA↔139↔0.44%	REC↔142↔0.44%	DOS↔148↔0.46%
IEN↔155↔0.49%	PRE↔155↔0.49%	NTA↔157↔0.49%	PRO↔170↔0.53%
ACI↔174↔0.54%	CIO↔176↔0.55%	ION↔184↔0.58%	RES↔184↔0.58%
MOS↔194↔0.61%	ICA↔231↔0.72%	ADA↔240↔0.75%	MEN↔246↔0.77%
CON↔281↔0.88%	ADO↔283↔0.89%	NTE↔406↔1.27%	ENT↔499↔1.56%

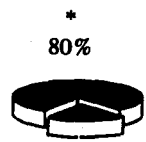
Quizá la primera conjetura que se nos ocurre pensar, luego de haber revisado este pequeño estudio, es que los monogramas y los bigramas no posicionales, no son capaces de realizar una apropiada detección de los errores ortográficos de la lengua hispana, salvo en el caso de que el texto que se quisiera revisar, estuviera virtualmente saturado de errores. De lo contrario, unas cuantas faltas de ortografía, digamos no más de cincuenta supresiones o transposiciones inválidas, no serían suficientes como para variar, en cantidades significativas, las frecuencias de aparición de los n-gramas, de modo que los sucesiones de letras derivadas de fuentes correctas e incorrectas, resultarían ser prácticamente iguales, no pudiendo entonces detectar la presencia de los errores. Tal vez este razonamiento, cuando se analiza en gran escala, sea el que motive a los investigadores a decir que los n-gramas nunca deben de ser aplicados aisladamente, sino siempre con el auxilio de alguna técnica de búsqueda en diccionario. Por lo que a los trigramas se refiere, hay que aceptar que parecen proporcionar una mejor eficiencia correctora que el resto de los n-gramas, en tanto que caracterizan más apropiadamente al léxico que los originó, pero, de cualquier forma, pensamos que un arreglo trigrama posicional sería más útil.

Concluímos esta exposición con la presentación de tres gráficas de frecuencia, todas ellas relativas a los n-gramas que estudiamos.

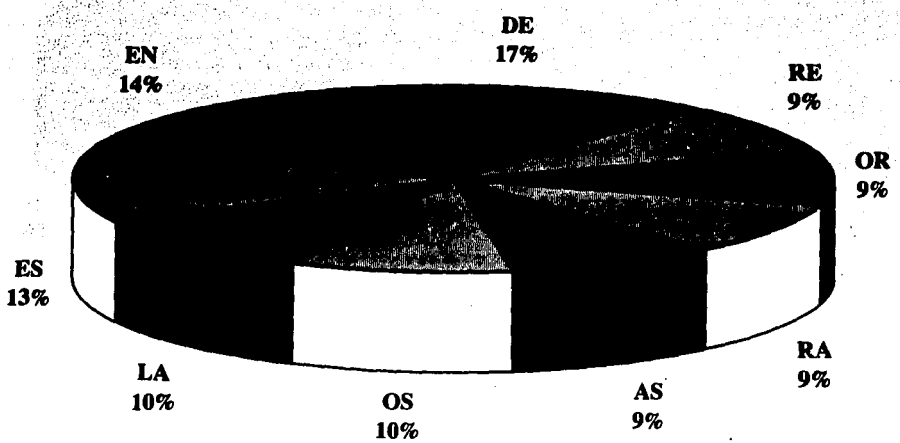
Frecuencia de los monogramas del español



Frecuencia de los bigramas del español



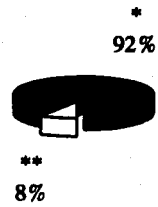
Los bigramas más frecuentes



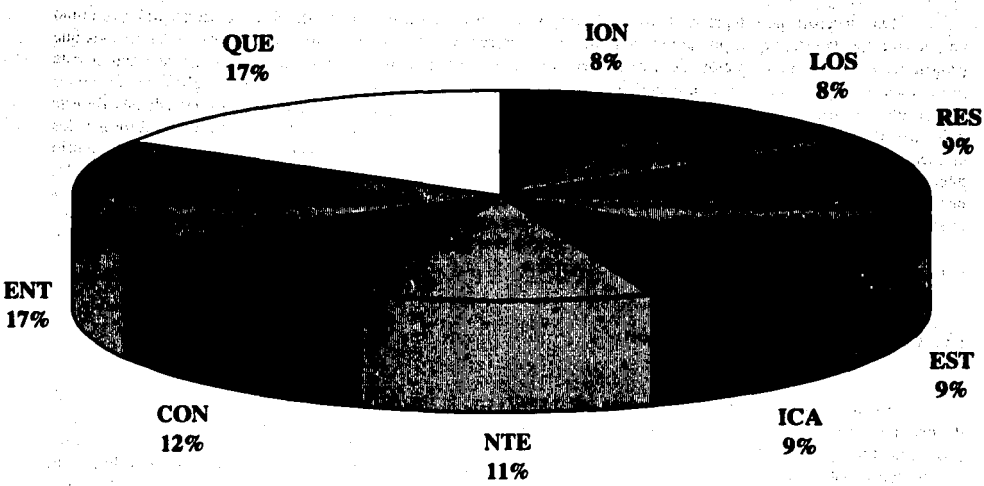
*** Total de bigramas que no forman parte de los nueve más frecuentes.**

**** Los nueve bigramas más frecuentes del español.**

Frecuencia de los trigramas del español



Los trigramas más frecuentes



*** Total de trigramas que no forman parte de los nueve más frecuentes.**

**** Los nueve trigramas más frecuentes del español.**

APÉNDICE 3
MANUAL DEL USUARIO

Esta sección fue integrada al final del cuerpo general de nuestro material de investigación y trabajo, como un apoyo a los usuarios que, por algún motivo, nos favorecerán con su interés por la aplicación de las rutinas que programamos. Con la intención de facilitar el conveniente manejo de los programas que acompañan a esta presentación, es prudente que este apéndice sea leído antes de ejecutar cualquiera de ellos, no obstante, lo menos que se podría esperar de nosotros es que nuestros diseños fueran "amigables", razón por la que, además de esta guía, los propios programas disponen de una opción que, en términos generales, expone los requerimientos y los elementos básicos de las diferentes etapas de su actividad. Dicha opción es llamada "Información", y el usuario puede conocerla a través de la tecla F1, siempre y cuando se encuentre posicionado en el Menú Principal. Aquellas personas que ya de por sí están familiarizadas con el ambiente de la detección ortográfica, podrán hacer uso de los programas sin necesidad de detenerse en este apéndice, mientras que aquellas que prefieran aprovechar una instrucción más elaborada, podrán iniciar su aprendizaje con la lectura de este manual frente a su computadora, al tiempo que realizan algunas pruebas con los programas.

§1. Las rutinas de programación

En total disponemos de seis rutinas relacionadas con la automatización ortográfica, aunque, en realidad, deberíamos de hablar de sólo cuatro, puesto que tres de ellas tienen una muy estrecha relación, y las razones de su programación son tan similares que bien podríamos comentarlas una sola. De cualquier modo, las contemos como seis o como cuatro, todas ellas son el resultado de nuestra investigación y objetivos de trabajo, y sus nombres son: *HAZ_DICC*, *1_MAIN*, *2_MAIN*, *3_MAIN*, *TABLA* y *REVISA*. Los programas *1_MAIN*, *2_MAIN* y *3_MAIN* son aquellos que bien podrían haber sido considerados uno solo, pues su objetivo es el mismo, salvo que cada uno realiza una tarea diferente. Así pues, mientras que *1_MAIN* se limita al cómputo de frecuencias de monogramas, *2_MAIN* hace lo propio con los bigramas, y *3_MAIN* con los trigramas. Por su parte, *TABLA* y *REVISA* son la pareja fundamental de nuestra tarea de programación, pues son los encargados de realizar la actividad de detectar los errores ortográficos, y las ambigüedades existentes en un texto determinado, lo cual constituye, como ya sabemos, el segundo y tercero objetivos del trabajo que aquí nos ocupa. Finalmente, *HAZ_DICC* es un programa de prueba y servicio para situaciones especiales. Originalmente, *HAZ_DICC* no estuvo contemplado dentro del trabajo, pero, por experiencia, hemos aceptado que su ejecución puede llegar a ser muy útil, según las intenciones que cada usuario pretenda.

Procedamos pues con las condiciones que deben ser cumplidas, forzosamente, por todos aquellos que desean favorecernos con el uso de nuestras rutinas.

§2. Requerimientos generales

Para poder ejecutar cualquiera de los programas que esta investigación generó, es indispensable contar con una máquina computadora de procesador 80286, como mínimo, pues aunque en principio ningún programa sería rechazado por un procesador de menor nivel, sí tendríamos que aceptar que su rendimiento, en cuanto a eficiencia y rapidez, se vería notablemente disminuido. En este sentido, no está de más advertirle a los usuarios que si desde antes de instalar nuestros programas, ya tienen problemas para trabajar con su software actual, quizá porque su computadora no dispone de suficiente memoria, entonces tendrá que instalar, físicamente, memoria adicional, si es que quiere conocer nuestro trabajo, o bien, tendrá que optimizar los recursos de memoria del equipo que posee. Por lo que se refiere al disco duro, es necesario que su máquina cuente con un mínimo de cinco megabytes libres, para garantizar la preservación de todos los archivos involucrados con las rutinas de detección ortográfica. Vale la pena mencionar que, a este respecto, el diccionario que será usado para verificar la correcta ortografía de cualquier archivo, ocupará, necesariamente, 2.5 megabytes, razón por la que el programa *REVISA* nunca podrá ejecutarse con menos de esa cantidad de bytes libres en disco.

Otro requerimiento infranqueable para el trabajo confiable de nuestras rutinas, es la necesidad de contar con un monitor a color, debido a que todos los programas mantienen una amplia interacción con el usuario, misma que fue implementada a través de cuadros de diálogo y mensajes en color, los cuales no podrán observarse en monitores blanco y negro. Inclusive, una vez que el programa detecta un dispositivo en blanco y negro, su ejecución podría llegar a suspenderse intempestivamente.

Las especificaciones requisitadas por nuestros programas, no sólo se limitan a aspectos de hardware, sino también a la manipulación de los archivos involucrados con su funcionamiento. Por razones de conveniencia, decidimos que nuestro detector ortográfico aceptara, exclusivamente, la lectura de archivos de texto standard, por ser estos los más accesibles para cualquier tipo y nivel de usuario. Un procesador de textos como *WordPerfect (versión 6.0)*, reconoce a estos archivos con el nombre de *textos ASCII*, mientras que para otros procesadores, como por ejemplo *Microsoft Word (versión 6.0)*, estos archivos se conocen como *textos MS-DOS*. Definitivamente, manejar estos archivos no debe de ser gravoso para nadie, antes al contrario, debe de resultar muy apropiado para todos, puesto que, actualmente, por razones de portabilidad, todos los procesadores de palabras nos permiten almacenar archivos en modo texto. Además, por la misma razón, cualquier procesador es capaz de leer este tipo de archivos, de modo que los usuarios podrán formatearlos luego de haberlos procesado con nuestras rutinas.

Un último aspecto importante de discutir en este punto, es el relativo a los archivos devueltos por los diferentes programas. Nuestras rutinas producen dos tipos diferentes de archivos: los de resultados y los que servirán para futuras ejecuciones. Los archivos de resultados siempre serán archivos de texto, que pueden ser leídos por cualquier editor, y que fueron hechos para que los usuarios guarden un reporte del resultado de la ejecución de sus programas. Por ejemplo, en el caso del programa *HAZ_DICC*, que sirve para crear un diccionario en modo texto, el archivo de resultados está formado, justamente, por el diccionario. En el caso de los programas que contabilizan n-gramas, el archivo de resultados está integrado por la cuenta de los n-gramas, y análogamente con el resto de las rutinas. El otro tipo de archivos devueltos, el de los que serán usados en posteriores ejecuciones, siempre serán binarios, de modo que el usuario no podrá disponer de un acceso cómodo a ellos, al menos no a través de los medios convencionales. Este tipo de archivos, como su nombre lo indica, sirve para guardar un registro de algunos datos que podría necesitar el programa para obtener reportes de próximas ejecuciones, o para mantener el control sobre la información que siempre estará requiriendo. Por ejemplo, en el caso de las rutinas de n-gramas, el archivo binario que se genera es usado para llevar la cuenta de las apariciones de los n-gramas que se han ido sumando a lo largo de varias ejecuciones, hechas sobre distintas entradas. No todos los programas

producen este tipo de archivos, pero lo recomendable es "no tocarlos" nunca, para permitir que cada rutina los procese a su propia conveniencia.

Por razones obvias, si no se cumple estrictamente con todas las condiciones que hasta aquí hemos especificado, la ejecución correcta de nuestras rutinas no está garantizada. Ahora bien, para aclarar dudas acerca del funcionamiento específico de cada programa, toca la oportunidad de platicar sobre cada rutina en especial.

§3. Las rutinas de n-gramas

A partir de este momento, relataremos las características particulares de cada programa, de modo que ustedes puedan conocer, paso a paso, qué es lo que va a ir haciendo la computadora, cuáles son los mensajes que aparecerán en la pantalla, y cómo deben ser interpretados.

Cuando los programas *1_MAIN*, *2_MAIN* y *3_MAIN* son arrancados, la máquina, como en todas las demás rutinas, presenta una carátula con datos elementales sobre el programador y el sistema, después de lo cual solicitará el nombre de un archivo de texto sobre el que se realizará el cálculo de la frecuencia de los n-gramas. Como sus nombres lo indican, *1_MAIN* cuenta monogramas, *2_MAIN* cuenta bigramas y *3_MAIN* cuenta trigramas. Al usar la palabra contar, pretendemos hacer referencia al hecho de que el programa tiene que ser capaz reconocer a todos los n-gramas existentes en el texto de entrada, al mismo tiempo que tiene que guardar la cuenta del número de apariciones que cada uno de ellos va llevando, para que al final pueda indicarle al usuario qué n-gramas aparecen en su texto, y cuántas apariciones tiene cada uno. Más aún, el programa tiene que poder sumarle a la cuenta de n-gramas del archivo en uso, la cuenta obtenida para archivos de anteriores ejecuciones, si es que ese es el deseo del usuario. Durante la actividad del sistema se reporta constantemente el porcentaje de avance en la lectura del archivo, y cerca del final, se advierte el porcentaje de avance en la escritura del archivo de resultados. El programa no establece límites en cuanto al tamaño del texto que será recibido, pudiendo, por lo tanto, procesar incluso un gran número de palabras. Ejemplos del archivo de resultados que genera cada programa aparecen a continuación. Nótese que una de las ventajas de producir archivos de texto standard, es que luego esos archivos pueden ser formateados tan arbitrariamente como puede observarse.

Ejemplo de la Salida del Programa 1_MAIN

Y-1621	V-2103	F-2239	ó-2356	Q-2541	G-3034
B-3682	M-7416	P-7879	U-10347	T-13577	L-14051
D-14455	C-15229	Y-18299	N-19690	R-20279	S-21131

Ejemplo de la Salida del Programa 2_MAIN

NC-1051	*GR-1058*	*OM-1081*	*EM-1084*	*LI-1095*
MO-1244	*ID-1259*	*IS-1267*	*RR-1318*	*AM-1332*
NO-1430	*ME-1503*	*PR-1521*	*AB-1533*	*LE-1567*
TR-1708	*RI-1710*	*MA-1747*	*lo-1780*	*ón-1795*

Ejemplo de la Salida del Programa 3_MAIN

NAR→520	**NTA→520**	**ONE→527**	**RRO→538**	**ERA→539**	**DIC→541**
DOS→548	**ERR→551**	**GRA→553**	**IST→554**	**RIO→565**	**RTO→566**
ARI→579	**ORE→587**	**STE→600**	**ORT→602**	**ORR→633**	**DEL→636**

Antes de solicitar el nombre del archivo que será leído, el programa pregunta si nuestro deseo es utilizar información preprocesada. Es en ese instante cuando la existencia de archivos binarios para uso de futuras ejecuciones toma sentido. La primera vez que usted ejecute alguna de nuestras rutinas de n-gramas, no puede pedirle al sistema que utilice información preprocesada, pues ningún archivo binario para tal efecto ha sido creado. Sin embargo, durante la primera ejecución, dicho archivo será generado, y a partir de entonces podrá usar esa información tantas veces como quiera. En adelante, lo único que tendrá que vigilar siempre que usted quiera emplear información preprocesada, es que exista el archivo *N_GRAMA.BIN* en el directorio raíz (en realidad se llamará *1_GRAMA.BIN* para el caso del programa *1_MAIN*, *2_GRAMA.BIN* para el caso del programa *2_MAIN* y, análogamente, *3_GRAMA.BIN* para el programa *3_MAIN*).

§4. El programa *HAZ_DICC*

HAZ_DICC es un programa que se encarga de recibir un archivo de texto cualquiera, y extraer de él todas las palabras que lo forman, con el fin de devolver un nuevo archivo de texto que contenga a dichas palabras, ordenadas lexicográficamente y observando que ninguna de ellas se repita. Originalmente se pensó que este programa, con algunas pequeñas modificaciones, construyera, y en su momento proporcionara mantenimiento, al diccionario que más tarde sería usado para las labores de detección ortográfica. Sin embargo, para esa actividad en particular, este programa no tardó mucho en demostrar su ineficacia. La razón principal de esto es que la rutina de mantenimiento del diccionario, tiene que proporcionar una forma práctica de especificar cuáles son las palabras que el usuario considerará como ambiguas, situación que no fue contemplada al programar *HAZ_DICC*. No obstante, ésta rutina es de gran utilidad cuando conviene tener a la mano una lista de las palabras que conforman un texto, por ejemplo, para verificar la cantidad de vocablos diferentes que existen en un documento, el tipo de léxico que es manejado por el escritor, o bien, simplemente, para disponer de un pequeño diccionario almacenado en la computadora de un modo que permita un acceso sencillo a su contenido. Aunque no lo parezca, esto último puede ser mucha utilidad para los usuarios comunes de los procesadores de textos comerciales que sepan explotarlo. En efecto, la mayoría de estos paquetes permiten agregarle palabras diferentes a los diccionarios que ya vienen incluidos en su instalación, y, normalmente, la forma de hacerlo es registrando palabras en un archivo que el paquete reconoce como diccionario alternativo. Esta ventaja suele ser aprovechada por aquellos que pretenden hacer uso del procesador para la creación de documentos escritos en lenguaje técnico. Nuestro programa *HAZ_DICC* puede facilitarle este trabajo a quienes lo tienen que hacer. Bastaría que el interesado hiciera correr *HAZ_DICC* con uno de sus documentos, para que éste se encargara de extraer de él todas sus palabras, y colocarlas después en un nuevo archivo, en el que estarían ordenadas y sin repeticiones. Más tarde este archivo podría servir para alimentar el diccionario del procesador en cuestión, y su problema quedaría corregido sin tener que escribir, una por una, todas las palabras que le interesaban.

§5. Los programas *TABLA* y *REVISA*

Como lo dijimos al principio de este apéndice, *TABLA* y *REVISA* son los pilares fundamentales de nuestro trabajo de programación, pues con ellos se crea y se mantiene el diccionario que se usa para detectar errores ortográficos, al mismo tiempo que se realiza la revisión ortográfica en sí.

Entender el funcionamiento de *TABLA* y *REVISA* es bastante elemental, sobre todo para quienes han atendido a las explicaciones que hemos venido presentando. En esencia, *TABLA* sólo le hace un par de preguntas al usuario en cada ejecución, la primera con el objeto de verificar si es que desea crear un nuevo diccionario, pregunta que tendrá que responder afirmativamente la primera vez que haga uso del programa, y también tantas

veces como elimine el archivo que almacena al diccionario. La segunda pregunta del programa sirve para averiguar el nombre del archivo que quiere usar alimentar al diccionario. Cualquier archivo de texto es útil para este fin. No se necesita que el archivo no repita ninguna de sus palabras, pues el mismo programa, en su rutina de creación y mantenimiento del diccionario, está capacitado para reconocer palabras duplicadas y eliminar todas las repeticiones de un vocablo, salvo la primera. Tampoco se requiere que el archivo posea un formato específico, como por ejemplo cada palabra en un renglón, o cada palabra escrita con mayúsculas. De este modo el usuario puede, simplemente, seleccionar algún archivo de texto que conozca que carece de errores ortográficos, y todas las palabras de ese archivo, sin importar su organización y formato, serán introducidas sin mayor problema en su diccionario. La opción que tiene que seleccionar para iniciar la creación o el mantenimiento del diccionario, es "Crear Tabla de Dispersión".

A las personas a quienes les basta con revisar la ortografía de algunos textos, no les hace falta saber nada más que esto, al menos por lo que se refiere al programa *TABLA*. No obstante, recordemos que otro de los objetivos de la tesis está relacionado con la detección de ambigüedades, y para que el programa *REVISA* pueda realizar este trabajo, es necesario que *TABLA* integre el diccionario de forma tal que sea posible distinguir entre una palabra común y una ambigüedad. Así pues, una vez que uno le indica al programa el nombre del archivo que alimentará al diccionario, la computadora pregunta si en dicho archivo existen o no ambigüedades. Cuando el usuario responde que sí existen ambigüedades, el programa asume que todas las palabras del archivo son ambiguas, mientras que una respuesta negativa obliga al programa a asumir que no existe ninguna ambigüedad. Conviene aclarar que una vez que una palabra ingresa al diccionario, automáticamente se le añade un código de identificación, que la clasifica como ambigüedad o no ambigüedad, y semejante código no podrá ser alterado mientras que el diccionario se conserve. Luego entonces, si el usuario desea integrar a su léxico una palabra que anteriormente fue introducida con un código de identificación de ambigüedad distinto, *TABLA* preguntará si es que verdaderamente se desea modificar ese código. Para algunos usuarios, la importancia de estas rutinas radicará en su capacidad para detectar errores ortográficos, y no en el beneficio de determinar si uno u otro vocablo constituye una ambigüedad, razón por la que convendría que ese tipo de usuarios siempre contestara que los archivos que usa para crear sus diccionarios, no poseen ninguna ambigüedad, de modo que el programa *REVISA* nunca se preocupara por buscarlas.

En términos generales, *REVISA* es aún más fácil de manejar que *TABLA*, pues lo único que necesita es del nombre del archivo que se desea revisar, el nombre del archivo que almacenará un reporte de los errores ortográficos encontrados, y el nombre de un archivo que guardará las ambigüedades halladas. Al final de su ejecución el programa envía un mensaje de fin de tarea, en el que aparece el cómputo del número de errores existentes en el archivo que se revisó, y luego el usuario podrá consultar los archivos que solicitó con fines de reporte. El archivo de errores ortográficos tiene una vista similar a las siguiente

El error ortográfico TOMMA fue hallado en la línea :
cercano a, nosotros tomma su nombre de la diosa romana del amor.

El error ortográfico RAZON fue hallado en la línea :
Por una razon dessconocida, la rotación de Venus alrededor de

El error ortográfico DESSCONOCIDA fue hallado en la línea :
Por una razon dessconocida, la rotación de Venus alrededor de

El error ortográfico ALRREDEDOR fue hallado en la línea :
rotación alrrededor del Sol.

Un formato muy similar a este es el que caracteriza al archivo que reporta las ambigüedades. Cada palabra, ya sea error o ambigüedad, esta acompañada de un mensaje que la anuncia como error o ambigüedad, así como de la línea en la cual aparece. Es obvio que si ningún error o ambigüedad es hallado, los archivos de errores y

ambigüedades quedan vacíos. Por supuesto que el archivo que el usuario deseaba revisar, no sufrirá ninguna alteración después de que nuestro programa lo haya analizado en busca de errores.

Cualquier palabra que no haya sido integrada en el diccionario a través del programa ***TABLA***, será considerada un error ortográfico, independiente de que en realidad sea un vocablo correcto o no. De igual forma, cualquier error ortográfico que se introducido a nuestro diccionario, será considerado como válido por el programa ***REVISA***, así que el usuario deberá de asumir la responsabilidad de vigilar todas las palabras que entran en su diccionario, por bien de sus futuras revisiones ortográficas.

Después de haber leído este manual del usuario, nadie debería de tener problemas para manejar a nuestros programas, y, de hecho, lo único que podría hacer falta para completar esta instrucción sería que el propio interesado practicara con su computadora. De cualquier modo, y por cualquier contrariedad que pudiera presentarse, reiteramos que todos los programas incluyen una opción de "Información", a la cual es posible acceder con sólo presionar la tecla F1 en el Menú Principal.

BIBLIOGRAFÍA

Magidin Matluk, Mario, *Estructuras de datos: algoritmos para sistemas de cómputo*, México, Ed. Trillas, 1991.

Pequeño Larousse ilustrado, Colombia, Ed. Printer Colombiana S.A., 1993.

Petersen, Richard, *Introductory C: pointer, functions and files*, U.S.A., Ed. Academic Press, Inc., 1992.

Peterson, J. L., *Computer program for detecting and correcting spelling errors*, A.C.M. Computing Surveys, Vol. 23, No. 12, U.S.A., December 1980, pp. 676-684

Kukich, Karen, *Techniques for automatically correcting words in text*, A.C.M. Computing Surveys, Vol. 24, No. 4, U.S.A., December 1992, pp. 377-439.

Kukich, Karen, *A comparision of some novel and traditional lexical distance metrics for spelling correction*, Proceedings of INNC-90-Paris, Paris, France, July 1990, pp. 309-313.

Schildt, Herbert, *C: Manual de referencia*, España, Ed. McGraw-Hill/Interamericana de España, S.A., Segunda edición, 1991.