Procesamiento del Lenguaje Natural, Revista nº 45, septiembre 2010, pp 137-144 recibido 30-04-10 revisado 16-05-10 aceptado 20-05-10

Errores ortográficos y de competencia en textos de la web en euskera

Orthographic and competence errors in the Basque Web

Iñaki Alegria, Izaskun Etxeberria
IXA taldea. Euskal Herriko Unibertsitatea
649 Postakutxa. 20080 Donostia
izaskun.etxeberria@ehu.es

Igor Leturia
Elhuyar Fundazioa
Zelai Haundi kalea 3. 20170 Usurbi
i.leturia@elhuyar.com

Resumen: En este trabajo se estima la calidad de los corpus en euskera obtenidos de la Web siguiendo una metodología similar a la propuesta por Ringlstetter et al. (2006) para el inglés y el alemán. Sin embargo nuestro trabajo difiere del mencionado en que al tratar un idioma de gran riqueza morfológica hemos optado por reutilizar verificadores ortográficos para reconocer los errores. Esto trae consigo, en nuestra opinión, una cobertura mayor de los errores que se estudian, además de la reutilización de recursos previamente desarrollados, lo que hace el método interesante para aplicarlo, sin prácticamente trabajo manual, a lenguas que tienen disponibles estos recursos. Los resultados van a ser de gran interés para detectar los distintos tipos de textos obtenidos de la Web en euskera según su corrección, y filtrar aquellos que pueden generar problemas o no tienen una calidad mínima.
Palabras clave: web as a corpus, errores ortográficos y de competencia, OCR, euskera

Abstract: The objective of the work presented in this paper is to estimate the quality of corpora retrieved from the Basque Web. The methodology i followed is similar to that used for English and Germany by Ringlstetter et al. (2006). The main difference lies in the fact that we reuse spelling checkers for detecting errors. We think that by this way we obtain a higher error coverage and that the method can be applied to other languages with practically no manual work provided such tools are available for them. The results obtained can be useful for improving the quality of corpora obtained from the web, eliminating documents containing errors over a given threshold.
Keywords: web as a corpus, spelling, competence errors, OCR, Basque

1 Introducción, objetivos y trabajos relacionados

El análisis de errores de grandes corpus tiene muchas utilidades. Entre las más destacadas está la selección de corpus de calidad suficiente cuando se usa la web como un corpus (*web as a corpus* - WAC) para extraer evidencias lingüísticas, colocaciones, terminología, etc.; ya que si la calidad de los corpus no está garantizada los resultados pueden no ser lo adecuados que se esperaba.

Cuando se obtienen automáticamente grandes cantidades de texto siempre hay una parte de los mismos que da problemas por distintas razones: formateado, conversión, transcripción, uso dialectal, coloquial o incorrecto, intercalación de textos en otros idiomas o de distinto registros, identificación incorrecta del idioma, etc. En la sección 4.4 del trabajo de Kilgarriff y Grefenstette (2009) hay ejemplos de algunos de estos problemas.

Este trabajo se enmarca en las tareas de usar la web como una fuente de datos abundantes y actuales para el estudio y desarrollo de herramientas para, en este caso, el euskera. Los datos de experimentación se basan en los trabajos anteriores de Leturia et al. (2008). El objetivo es poder detectar los distintos tipos de textos según su corrección y filtrar aquellos que pueden generar problemas.

La metodología que utilizamos es similar a la utilizada por Ringlstetter et al. (2006) en su interesante aportación. En este trabajo ellos

ISSN 1135-5948

estudian la distribución de los errores ortográficos de varios tipos de paginas web en inglés y alemán y, entre otras conclusiones calculan los umbrales de las tasas de error para clasificar los documentos como excelentes, buenos, malos y descartables. Distinguen dos tipos de corpus (generales y específicos), dos tipos de documentos (HTML y pdf) y estudian cuatro tipos de errores: tipográficos, de codificación, cognitivos y OCR.

Tras diseñar y poner en marcha un sistema de *crawling* para obtener las muestras que les interesan para cada tipo de textos, tokenizan y eliminan palabras conflictivas (palabras con mayúsculas, cifras, palabras cortas...). Paralelamente generan diccionarios de errores para cada tipo de error, trabajo que hacen de forma semiautomática. Finalmente cuentan el número de errores de cada tipo que aparecen en cada documento, hacen estadísticas y obtienen las características de cada clase de documento.

Tomando como referencia esta metodología hemos diseñado un sistema con los mismos objetivos para la web en euskera, pero con ciertas diferencias remarcables:

- El trabajo de *crawling* se evita reutilizando trabajos anteriores (Leturia et al., 2008). A partir de esos corpus se obtiene una muestra al azar para cada tipo de corpus y documento (ver sección 3).
- No se utiliza una lista de errores sino correctores basados en estados finitos que se habían desarrollado anteriormente (sección 2). Al tratar un idioma de gran riqueza morfológica creemos que esta opción es más adecuada, aunque en ciertos casos tomaremos como errores algunas palabras correctas que no son reconocidas como tales (p. ej. neologismos que se parecen a errores de palabras existentes). Sobre esto se discute en la sección 3.2.
- Se quieren estimar los errores tipográficos, cognitivos y OCR, pero los de codificación quedan fuera de nuestros objetivos ya que no se han detectado problemas especiales en ese sentido.
- En un futuro se quiere completar el estudio con la detección de textos en euskera dialectal y diacrónico.

Otro tema relacionado interesante es usar la web para detectar y corregir errores (Whitelaw et al., 2009).

A continuación, en la sección 2, introducimos la morfología de estados finitos y su utilización para generar correctores. Posteriormente, en la sección 3, se describen el entorno experimental y los resultados obtenidos, y finalmente en la sección 4 se presentan las conclusiones y los trabajos planificados para el futuro.

2 Morfología de estados finitos y correctores

En esta sección se presentan los distintos analizadores/verificadores utilizados precedidos por una revisión de la tecnología en que están basados y de la herramienta utilizada.

2.1 Tecnología

Un procesador morfológico es una herramienta básica para el PLN. Si el idioma es de flexión rica una lista de palabras con su análisis correspondiente (como la que se usa en ciertas aplicaciones para ciertos idiomas) no es una solución adecuada.

La tecnología de estados finitos ha sido aplicada exitosamente para el desarrollo de analizadores y generadores morfológicos. La morfología se describe por medio de dos ficheros, (1) el léxico, donde se describen los morfemas y los conjuntos de morfemas que les pueden seguir (morfotáctica); (2) las reglas fonológicas, que describen los cambios producidos al encadenar los morfemas (Beesley & Karttunen, 2009). Todos los elementos se compilan en transductores que pueden componerse en uno solo y tanto el análisis como la generación de palabras se realiza a gran velocidad.

Las reglas fonológicas pueden ser paralelas o secuenciales (Alegria et al., 2009). Las paralelas tienen la ventaja de que el orden no es significativo y que no hace falta definir lenguajes intermedios entre las palabras del texto (nivel superficial) y la representación léxica (nivel léxico). Sin embargo estas reglas deben tener en cuenta en sus contextos los efectos de las reglas relacionadas, y esto muchas veces es fuente de errores. Ambos tipos de reglas pueden ser compiladas y convertidas a transductores muy eficientes. Estos

transductores son los datos de los procesadores morfológicos correspondientes, que sirven tanto para análisis como para generación.

Se han desarrollado procesadores morfológicos basados en esta tecnología para gran cantidad de idiomas, incluyendo finés, euskera, turco, zulú e inglés. Esta tecnología se ha aplicado especialmente a los idiomas morfológicamente ricos, ya que las soluciones más simples que se utilizaban anteriormente no eran adecuadas para ellos.

Si se dispone de un procesador morfológico la construcción de un corrector ortográfico es casi inmediata: las palabras correctas son las que pueden ser analizadas (Kukich, 1992) (Alegria et al., 2002).

2.2 Herramientas

El toolkit de Xerox (Beesley & Karttunen, 2009) proporciona herramientas para reglas paralelas (*twolc*) y reglas secuenciales (*xfst*), además de para el léxico (*lexc*). Su experiencia indica que el uso de reglas secuenciales resulta a la larga más cómodo, tendencia que se ha visto confirmada en posteriores implementaciones.

Las herramientas de Xerox son de gran calidad y consiguen transductores muy compactos, pero tienen un gran inconveniente; la licencia es muy restrictiva y, salvo excepciones, no puede ser usada para aplicaciones comerciales. Ante ello se han desarrollado en software libre herramientas que pueden sustituir las de Xerox. Queremos destacar dos: *hunspell* y *foma*.

hunspell no trabaja con transductores, pero es una solución mejor que sus predecesores *ispell*, *aspell* y *myspell*, ya que permite mayor número de paradigmas y doble encadenamiento de sufijos. No acepta reglas paralelas ni secuenciales independientes del léxico, ya que los cambios deben ser explicitados en la descripción del léxico. Su mayor interés es que está aceptado por las últimas versiones de OOffice y Mozilla, por lo que una descripción para un idioma usando *hunspell* deviene automáticamente en un corrector ortográfico para estas herramientas.

foma (Hulden, 2009) es un toolkit que quiere ser equivalente a las herramientas *xfst* y *lexc* de Xerox pero con una implementación independiente y licenciado bajo GPL (*http://foma.sourceforge.net/*). Por lo tanto es adecuado para descripciones de morfología de estados finitos usando reglas secuenciales. Según su autor las descripciones para Xerox funcionan directamente, y la compilación es más eficiente. Tiene una ventaja adicional, y es que las descripciones preparadas para *lexc* y *xfst* de Xerox se compilan directamente en *foma* (salvo alguna excepción y siempre que los datos estén en Unicode)*;* ya que esta nueva herramienta integra los mismos comandos y la misma sintaxis que las herramientas citadas. Esto hace que el libro citado (Beesley & Karttunen, 2009) pueda ser usado en su mayor parte como un manual de *foma*.

2.3 Verificador ortográfico y detectores/correctores de errores

Para detectar las palabras correctas y los distintos tipos de errores (tipográficos, cognitivos y OCR) se han utilizado los siguientes recursos:

1. transductor del euskera estándar. Sirve para identificar las palabras correctas.
2. transductor para detección de errores de competencia
3. función *med* de *foma* sobre el transductor estándar. Sirve para obtener posibles errores tipográficos de adición, sustitución y eliminación.
4. transductor para detección de transposiciones
5. transductor para detección de errores OCR

Los vamos a describir uno a uno.

Transductor del euskera estándar

Usando tanto las herramientas de Xerox (Alegria et al., 2002), como *foma* (Alegria et al., 2009) se ha desarrollado un procesador para el euskera estándar y varios correctores.

El léxico contiene más de 80.000 entradas, la morfotáctica y la correspondiente información morfológica relacionada. Alrededor de 23 reglas complejas describen los cambios fonológicos usando reglas paralelas (Alegria et al., 2002) o secuenciales (Alegria et al., 2009). En el segundo artículo se describe el proceso de migración de unas a otras.

Transductor para errores de competencia

Además del transductor estándar se ha desarrollado uno ampliado (*enhanced*

transducer (Alegria et al., 2002)) con el objetivo de detectar, y en su caso corregir, errores de competencia y ciertas variantes dialectales.

Al léxico y a las reglas del euskera estándar, se añaden entradas léxicas adicionales (relacionadas con su forma estándar) y un sistema complementario de reglas no estándares. Así el sufijo dialectal *-tikan* es añadido al léxico relacionado con su forma estandar *-tik* (corresponde a la preposición *de*/*desde*).

La siguiente regla también es añadida entre un par de reglas de caracter principalmente fonológico:

```
h (->) 0 || _ ;
```

Esta regla indica que la *h* del léxico puede no aparecer en la palabra del texto por un error (para indicar que es opcional el operador `->` aparece entre paréntesis).

Con estas dos adiciones es posible reconocer como variante o error de competencia varias formas, por ejemplo *zuhaitzetikan*, *zuaitzetik* y *zuaitzetikan*, tres variantes de *zuhaitzetik* (*desde el árbol*)

Las reglas también han sido convertidas y probadas con *foma* (Alegria et al., 2009), y la composición secuencial ha permitido una mayor modularidad, con lo que no ha habido que hacer ningún cambio en las reglas estándares.

Con este transductor es posible, además de reconocer variantes, obtener la forma estándar correspondiente a las mismas (tanto para formas dialectales como para errores de competencia), y hasta saber las reglas que se han aplicado (Alegria et al., 2009). Esto es interesante además de para detección de errores para sistemas de aprendizaje de idiomas asistidos por ordenador.

Transductor OCR

Basándonos en experimentos anteriores y en las reglas conocidas de OCR hemos desarrollado un conjunto de reglas básicas para detectar transformaciones típicas que generan los programas de OCR cuando los resultados no son posteriormente revisados.

Unos 25 cambios del tipo *c*/*e*, *n*/*ri*, *rn*/*m* han sido descritos en simples reglas sin contexto en el formato de *foma*, pero usando reglas paralelas en este caso con el fin de limitar a uno el número de errores OCR por palabra. Estas reglas deben ser revisadas ya que constituyen una primera versión y los resultados no han sido evaluados exhaustivamente.

Dichas reglas se han compuesto con el transductor estándar con lo que es posible identificar un error OCR de los previstos en cualquier palabra estándar.

Función *med*

El tratamiento de los errores tipográficos se podría describir por medio de reglas foma/xfst pero la opción *med* de *foma* (Alegria et al., 2009) nos permite detectar, entre otros, errores tipográficos correspondientes a adición, sustitución y eliminación de un carácter (Kukich, 1992). Para detectar todos los errores tipográficos de distancia uno nos faltan los errores de transposición.

Transductor para transposiciones

Los errores de transposición (cambiar de orden dos caracteres consecutivos) los hemos descrito por medio de un conjunto de reglas *foma*, que hemos compuesto con el transductor estándar, con lo que conseguimos identificar cualquier transposición en las palabras estándar.

Por ejemplo para un par de caracteres contiguos que empiece por *a*, una primera regla marca el cambio por medio de un símbolo especial (p. ej. *1*) y duplica el primer carácter al final del bigrama (*Alf* representa una letra cualquiera).

```
"a" Alf (->) "1" ... "a" ;
```

Una segunda regla elimina el símbolo y el carácter que le sigue:

```
"1" Alf -> 0 ;
```

Componiendo las dos reglas el par de caracteres "ab" se transforma en "1aba" por la primera regla y en "ba" por la segunda, con lo que la palabra *kalbaaza* puede ser identificada como transposición de *kalabaza*.

3 Experimentos y resultados

Como hemos dicho en los objetivos de la sección 1, nuestra meta es estudiar la cantidad y naturaleza de los errores en los corpus en euskera obtenidos de la web, para en el futuro diseñar un filtro que garantice la calidad de los mismos.

3.1 Diseño experimental

Corpus

Para obtener una muestra representativa de la calidad de los corpus nos hemos basado en los trabajos previos de WAC en euskera. Al igual que en el trabajo referenciado (Ringlstetter et al., 2006) hemos distinguido dos tipos de corpus (generales y específicos) y dos tipos de documentos (HTML y PDF) con lo que tenemos 5 corpus.

El corpus general se generó a partir de una lista de las 500 palabras más comunes en euskera y enviando combinaciones aleatorias de tres de ellas a los buscadores (tal y como hace Sharoff (2006)), y sin realizar el filtrado de tema al final. Se obtuvo un corpus que consta de 71.500 documentos y 81 millones de palabras. A partir de este corpus se extrajo una muestra al azar de documentos HTML y PDF con las características que se especifican en la tabla 1. El número de documentos obtenidos es el mismo que en (Ringlstetter et al., 2006).

Tipo	Documentos	Palabras
HTML	2000	3,2 M
PDF	1000	2,5 M

Tabla 1.- Tamaño de la muestra del corpus general

Por otro lado los corpus específicos utilizados en los experimentos son una muestra obtenida al azar de los documentos obtenidos utilizando la metodología explicada en los trabajos de Leturia et al. (Leturia et al., 2008) en tres áreas: turismo, informática y biotecnología. En la tabla 2 se muestran los tamaños de cada uno de ellos.

Corpus	Documentos	Palabras
Turismo	1238	1,5 M
Informática	1672	2,5 M
Biotecnología	302	0,6 M

Tabla 2.- Tamaño de los corpus específicos recopilados antes del muestreo

De los documentos de cada tipo se hizo un muestreo al azar para obtener los documentos que se utilizan en los experimentos.

Las características se detallan en la tabla 3. En (Ringlstetter et al., 2006) también utilizan 500 documentos en las muestras de corpus específicos.

Corpus	Documentos	Palabras
Turismo	500	0,6 M
Informática	500	0,7 M
Biotecnología	302	0,4 M

Tabla 3.- Tamaño de la muestra de corpus específicos

Procesado

Para evitar tomar por errores un gran número de palabras que no lo son, tras el proceso de tokenización, por cada documento se hace un filtrado de las palabras cortas y de las que llevan mayúsculas o caracteres especiales, siguiendo las mismas directrices que (Ringlstetter et al., 2006).

Posteriormente utilizando el transductor estándar, en cada documento se dan por buenas aquellas palabras analizadas por el mismo. Posteriormente las palabras no reconocidas se pasan por las siguientes fases:

- Se analizan por medio del reconocedor/corrector de errores de competencia, obteniéndose las posibles palabras con este tipo de error.
- Se analizan usando el reconocedor/ corrector de errores OCR, para obtener así el listado de este tipo de errores .
- Utilizando la opción *med* con el transductor estándar y una distancia máxima de uno (sólo se admite un cambio) se obtienen los errores tipográficos de adición, sustitución y eliminación. Además usando el transductor de transposición se obtiene los errores de transposición. Estos errores se unifican en una única lista para evitar que una palabra que puede ser interpretada como transposición y sustitución por ejemplo se cuente dos veces. Además se eliminan las palabras que se han analizado como errores de competencia ya que muchos errores pueden tener doble interpretación y las reglas de los errores de competencia son más precisas.

Se debe resaltar que ciertas palabras listadas como errores no son tales, sobre todo en el caso de los errores tipográficos, donde puede ser bastante habitual que ciertas palabras correctas que no están en el léxico difieran en un carácter de otras que existen. Por ejemplo la forma *blogean* (*en el blog*) no es reconocida como forma correcta y se interpreta como un error tipográfico de la palabra *blokean* (*en el bloque*). Este problema también es estudiado en (Ringlstetter et al., 2006). Otro problema a mencionar es el que ciertas palabras escritas en otros idiomas se interpretan también como errores ortográficos, tanto por aparecer en el propio texto como por ser etiquetas HTML que no se han podido filtrar adecuadamente. Esto podría ser tratado por medio de un filtro al efecto, pero siendo el objetivo detectar corpus de una calidad mínima nos parece que considerarlos como errores también cumple su función.

Por lo dicho anteriormente se concluye que los llamados *real-word errors* (errores que generan palabras correctas) (Kukich, 1992) no se tendrán en cuenta, ni tampoco los tipográficos con más de un error en cada palabra, pero esto es lo habitual en este tipo de estudios.

3.2 Resultados

Los resultados para los distintos tipos de corpus y de documentos se presentan a continuación.

El la figura 1 y en la figura 2 se muestran los resultados de los tres tipos de errores (competencia, tipográficos y OCR) para la muestra del corpus general.

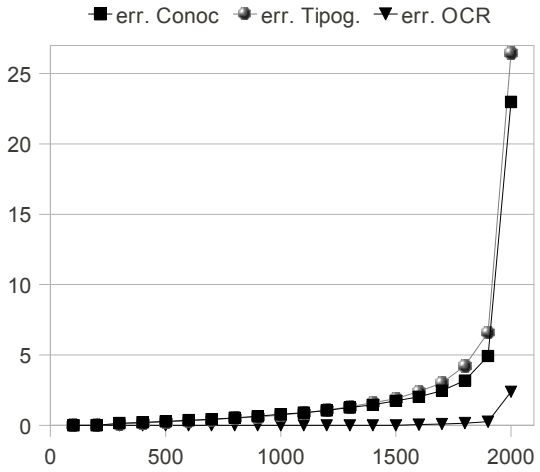

Fig 1.- Tasa de error en documentos HTML

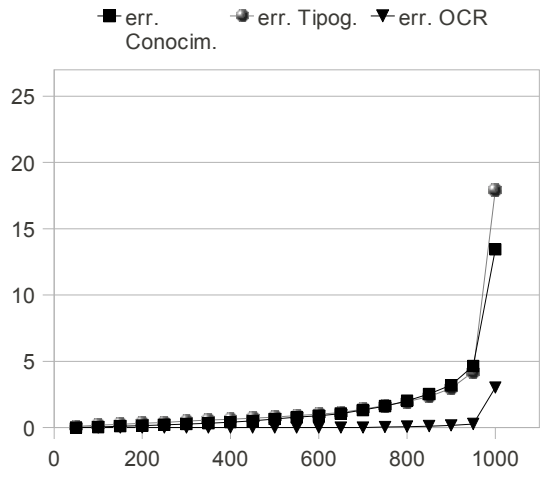

Fig 2.- Tasa de error en documentos PDF

En la figura 1 se reflejan los resultados para los documentos HTML y en la figura 2 para los de tipo PDF.

Examinando ambas gráficas podemos observar que los resultados son similares:

- Más de la mitad de los documentos no contienen prácticamente errores. Los errores se concentran en un 10% de los documentos.
- Los errores de competencia son comparables a los tipográficos. A la espera de un análisis más pormenorizado de los resultados esto confirma nuestra hipótesis de la influencia del uso dialectal y otras variantes de las formas estándares actuales.
- Los errores OCR son mucho menos numerosos que los de otro tipo. Contrariamente a lo que esperábamos la tasa de errores OCR es similar en ambos tipos de formatos, lo que se puede observar con más detalle en la figura 3. Esto parece indicar (y así lo hemos comprobado en una análisis previo buscando desviaciones OCR de las palabras más habituales en euskera) la mínima presencia de documentos escaneados sin verificar.

Reflejamos los datos obtenidos para el corpus de informática que son similares a los del corpus de biotecnología (menos representativo por el tamaño del corpus).

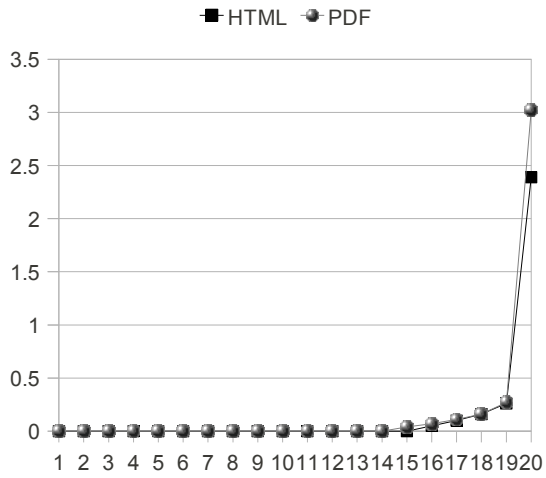

Fig 3.- Tasa de error OCR en documentos HTML y PDF

En la figura 4 se reflejan los mismos datos que en las figuras 1 y 2 pero para textos especializados.

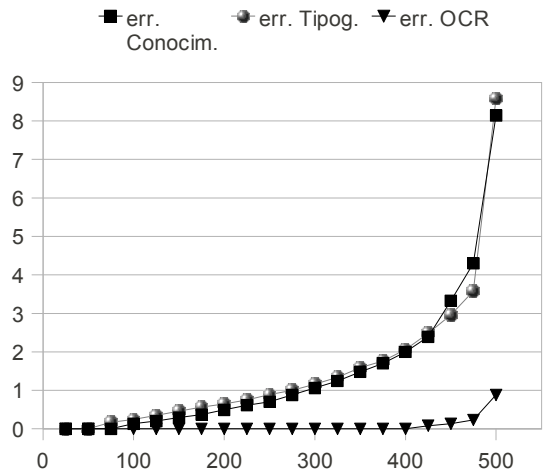

Fig 4.- Tasa de error en documentos de informática

Los de turismo tienen un comportamiento más similar al corpus general, lo que es normal ya que no se puede considerar un dominio técnico.

Comparando los errores en el corpus general con los de los documentos técnicos se confirma que los documentos técnicos contienen menos errores.

Sin embargo comparando los resultados para el euskera con los obtenidos para el inglés y alemán en (Ringlstetter et al., 2006) se detecta una mayor tasa de errores para el conjunto de los documentos, debido a la utilización de reconocedores/ analizadores genéricos y no de listas preparadas al efecto. A la espera de un estudio manual de los resultados, podemos decir que en el trabajo de referencia se conseguía más precisión en la detección de errores y en nuestro sistema se consigue mayor cobertura.

Antes de establecer una clasificación similar a la propuesta por Ringlstetter et al. (2006) (con porcentajes de textos clasificados como excelentes, buenos, regulares y malos) es necesario un estudio manual más detallado de los resultados obtenidos. Un problema que hemos detectado es la aparición de textos dialectales y diacrónicos que no encajan en ninguna de las categorías previstas (correcto, tipográfico, competencia y OCR). Estamos preparando un detector de textos escritos en euskera dialectal.

4 Conclusiones y trabajo futuro

En nuestro trabajo hemos tomado como base el trabajo de Ringlstetter et al. (2006) y como objeto de estudio y experimentación una muestra de los corpus recuperados en los trabajos de Leturia et al. (2008). Sin embargo nuestro trabajo difiere del mencionado en que al tratar un idioma de gran riqueza morfológica hemos optado por transductores para reconocer errores en lugar de listas de errores. Esto trae consigo, en nuestra opinión, una cobertura mayor de los errores que se estudian, además de la reutilización de recursos previamente desarrollados, lo que hace el método interesante para aplicarlo, sin prácticamente trabajo manual, a lenguas que tienen disponibles estos recursos.

Los resultados van a ser de gran interés para detectar los distintos tipos de textos obtenidos de la Web en euskera según su corrección, y filtrar aquellos que pueden generar problemas o no tienen una calidad mínima.

Agradecimientos

Proyecto parcialmente subvencionado por los proyectos OpenMT2 (Ministerio de Ciencia e Innovación, TIN2009-14675-C03-01) y Berbatek (Eusko Jaurlaritza, IE09-262). Gracias a Mans Hulden por su ayuda en la construcción de los transductores usando *foma*.

Bibliografía

Alegria I., Aranzabe M., Ezeiza A., Ezeiza N., Urizar R. 2002. Using Finite State Technology in Natural Language Processing of Basque. LNCS: Implementation and Application of Automata. 2002. Springer.

Alegria I., Etxeberria I., Hulden H., Maritxalar M. 2009. Porting Basque Morphological Grammars to foma, an Open-Source Tool. FSMNLP2009. Pretoria. South Africa.

Beesley K. R. and Karttunen L. 2003. Finite State Morphology. CSLI Publications, Palo Alto, CA.

Hulden M. 2009. Foma: a Finite-State Compiler and Library. EACL 2009. Demo session. pp 29-32.

Kilgarriff, A. and Grefenstette, G. 2003. Introduction to the special issue on the web as corpus. Computational linguistics, 29(3): 333-347. MIT Press.

Kukich K. 1992. Techniques for Automatically Correc-ting Words in Text. ACM Comput. Surv. 24(4): 377-439.

Leturia I., San Vicente I., Saralegi X. and Lopez de Lacalle M. 2008. Collecting Basque specialized corpora from the web: language-specific performance tweaks and improving topic precision. Proc. of the 4th. Web as Corpus Workshop. LREC 2008.

Ringlstetter, C. and Schulz, K.U. and Mihov, S. 2006. Orthographic errors in web pages: Toward cleaner web corpora. Computational Linguistics, 32(3): 295-340. MIT Press.

Sharoff, S. 2006. Creating General-Purpose Corpora Using Automated Search Engine Queries. WaCky! Working Papers on the Web as Corpus, 63-98. Ed. Marco Baroni and Silvia Bernardini. Bologna.

Whitelaw C., Hutchinson B., Chung, G.Y. and Ellis G. 2009. Using the web for language independent spellchecking and autocorrection. Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2, 890-899.