

# Using Google to Create a More Accurate and Easily- Extensible Spelling Corrector

---

July 19, 2008

*Hepburn Best*  
hlb0401@aim.com

*Hayden Metsky*  
hmetsky@gmail.com

*Jose Rosario*  
vrosario2@gmail.com

*Colin Sidoti*  
sidoti3291@gmail.com

*Jacqueline Somogyi*  
jackie@new-dork-city.net

Spell checkers are now a common, integrated part of many commercial and freely available word processing programs. Agglutinative languages (such as Hungarian and Finnish) pose a separate problem, as there are many different “correct” forms for any given word. Due to the seemingly infinite number of possible words, the limited scope of a dictionary (provided with most spell-checking software) poses an obvious problem – if not only in terms of computability – for completeness. In this paper, we explore means of isolating and recommending corrections for misspelled words in English using the Internet as a corpus, and then discuss methods of extending these processes to another language, specifically the agglutinative Hungarian.

## 1 Introduction

In high-school classrooms, it is joked that if a word is misspelled that the writer did not use the spell checker in his word processor. The necessity of spell-checking software has made it a common household phrase since the technology's creation in the late 1970s. From letters to loved ones, a resume for a job interview, or a research paper, a spell checker is a necessity to a successfully written document. From the most common misspelled words to the most scientific or complicated words in a language, a spell checker included in a word processor can prove helpful to a computer user of any proficiency level. Today, in this age of technology, the availability of an accurate and responsive spell checker is of the utmost importance.

## 2 Current Options

Three of the most popular spell checkers are the Microsoft Word spell checker, Aspell<sup>1</sup>, and Ispell<sup>2</sup>. According to their Suggestion Intelligence (SI) rating, which corresponds to the rate of good suggestions proposed by the spell checker with respect to all the misspellings of test files, all three programs have an accuracy rate of about 78 percent. However, according to their Suggestion Intelligence First (SIF), which represents the ability of the spell checker to find the good suggestion at the first position of the list of all the suggestions, Word scored the highest with an accuracy of 65%, followed by Aspell and Ispell with 58% and 39% accuracies, respectively. One possible explanation for this was that these spell checkers relied only on the word itself instead of on its context. [2] As shown by

this data, current spell checkers leave much room for improvement.

Many spell checkers rely on the traditional method of using dictionaries to isolate and correct misspelled words (with a common example of one that does not being the spell checker used in Word 2007<sup>3</sup>); the use of context in conjunction with traditional dictionary lookup as a metric used in spelling correction is a more recent advancement. This however has its flaws. For example, the English language continues to adopt new words, proper nouns, and foreign loan words into the language. In 2006, Google was adopted into both the Webster and Oxford English dictionaries (albeit as a common – “non-proper” – noun) (Arnfield); however, some spell-checking software included in word-processing applications does not recognize Google as a common noun. Also, the dictionaries for some languages are not as readily available as those for English would be. English is the *de facto* vernacular in the computing community, evident by the fact that approximately 71% of all web pages are primarily written in English [6]. For the sake of contrast it may be noted that a less common language such as Swedish is the primary language of only about 0.7% of the world's web pages [6].

We aim to improve on current spell checkers by using the Web as a corpus, extending work presented in [2]. Due to the Web's large size and rich diversity, it can be an effective frame of reference to which documents may be compared. For the research presented in this paper, we used a combination of the dynamic (non-cached) Web and corpora in different languages that we have assembled from the public domain document repository Project Gutenberg<sup>4</sup> (which serve as a static, cached portion of the web). These languages are English and

---

<sup>1</sup><http://www.gnu.org/software/aspell>

<sup>2</sup><http://ficus-www.cs.ucla.edu/geoff/ispell>

---

<sup>3</sup><http://www.microsoft.com/word>

<sup>4</sup>[http://www.gutenberg.org/wiki/Main\\_Page](http://www.gutenberg.org/wiki/Main_Page)

Hungarian. Hungarian was added as a language of interest because we also strive to create an accurate spell checker that can detect and suggest corrections for agglutinative words.

Using the search engines Google<sup>5</sup>, we will determine the appropriate correction for a misspelled word by analyzing the number of times each generated replacement appears on the Web. We will also use context, taking into account the instances the word appears on the Web in phrases identical to those in which the correction word must appear in the document. The list of generated corrections is first populated by finding all the words within a certain number of Damerau-Levenshtein edit-distances from the misspelled word. It is then reduced in size by finding which of these generated words include n-graphs (on the letter scope) which do not appear in the target language. Finally, the trimmed list is compared to a dictionary file, and only words found in the dictionary are passed on to the search engine in a phrase query. We will also use keyboard distance as a metric for determining the correct replacement, under the assumption that the correct letter a person meant to type is likely a short distance away from the letter he actually typed on the keyboard. By combining these methods, we aim to create a more accurate spell checker that is easily extensible to other languages as well. In addition to other languages, it can hopefully be used to correct to slang and other neologisms in a language.

### 3 Background Information

A *spell checker* is a program that identifies misspelled words in a document and attempts to correct them. Typically, it will compare each word against a dictionary: if the word is found in the dictionary it is considered correctly spelled, and the

---

<sup>5</sup><http://www.google.com>

program will move on to the next word. However, if the word is spelled incorrectly (which is the assumption made by the program if the word cannot be found in the dictionary) the program will try to suggest a word that it thinks the user intended to write.

#### 3.1 History of the Spell Checker

Maria Mariani, an English for Heritage Language Speaker Program Manager at Georgetown University<sup>6</sup>, and five other linguists from the same university developed the first spell checker for IBM in the 1970s. Spell checkers for personal computers first appeared for CP/M and TRS-80 computers in 1980, and were used more widely with the introduction of the IBM PC in 1981. In the beginning, PC's allowed for spell checkers but as a standalone program. These programs required a large amount of energy, which proved to be a drawback for the original spell checkers[3].

By the mid 1980s many popular word-processing applications (such as WordStar and WordPerfect) had integrated spell checkers. Concurrently spell checkers extended from only English into non-Latin alphabet languages. However, correcting such languages required increasing sophistication in the morphology-processing routines of the software (which languages such as Hungarian also require)[3].

In recent years, spell checkers have become increasingly prolific. Notably, Firefox 2.0<sup>7</sup> was the first major Internet browser that allowed for passive spell-checking of the content that the user entered into a text box. This then prompted the web browsers Konqueror<sup>8</sup> and Opera<sup>9</sup>, the email service Gmail<sup>10</sup> and the instant messaging client Pidgin<sup>11</sup> to also offer spell checking support;

---

<sup>6</sup><http://cled.georgetown.edu/faculty/#mariani>

<sup>7</sup><http://www.mozilla.com/en-US/firefox/features/>

<sup>8</sup><http://www.konqueror.org/>

<sup>9</sup><http://www.opera.com>

<sup>10</sup><http://mail.google.com>

<sup>11</sup><http://www.pidgin.im>

all use Aspell as their spell-checking engine. The Apple operating systems now have integrated spell-checking ability in almost all applications included with the OS<sup>12</sup>[3].

## 3.2 Languages

### 3.2.1 English

English is the primary language of our spell checker. It is the most commonly spoken language in the world, with about 1.8 billion speakers. It is the dominant international language in communications, science, business, entertainment, and diplomacy, and is one of the six official languages of the United Nations.

English is part of the West Germanic branch of the Indo-European languages, but has minimal inflection compared with other languages in the group. It lacks grammatical gender and adjectival agreement. The language has changed considerably through the ages, with case marking and the remnants of inflection disappearing over time. Most English words can trace their origins to other languages; for example, about 28% of words have Latin roots, another 28% are from French, and 25% are from other Germanic languages[8].

### 3.2.2 Hungarian

Hungarian is spoken by 14.5 million people in Hungary and in parts of Romania, Slovakia, Serbia, Ukraine, Croatia, Slovenia, and Austria. It has long been of interest to linguists because it is one of very few modern European languages that do not belong to the Indo-European language group. Part of the Finno-Uralic language family, it can sound completely foreign to speakers of other European languages, and is considered one of the most difficult languages to learn[4].

Hungarian is an example of an *agglutinative language*, in which most

words are formed through the joining of morphemes, the smallest meaningful units of a language. For example: to pluralize, form the diminutive, or change tense, the root words remain unchanged. Instead, one or more affixes are added to the word[7].

These languages have often been very difficult for spell checkers to analyze. Therefore, we will test our spell checker on a Hungarian document in order to measure its effectiveness.

## 4 Potential Problems

One problem that we immediately foresee is the possible denial of access to Google. We suspect that there could be a limit to how many times the search engine could be accessed by a particular IP address in a certain amount of time. Another possible problem is whether a fair comparison between the corpora that we manually obtained from the Internet and the search engines themselves can be made.

## 5 Related Work

The purpose of our project is to come up with a better way to be able to suggest a correct term when a word is misspelled. Our answer is “the Internet.” However, this idea actually comes from a research group in 2007 which included Stéphanie Jacquemont, François Jacquenet, and Marc Sebban[CITATION NUMBER]. The purpose for their research was to find out whether the Internet, specifically the search engine Google, could be used as a corpus for a spell checker, which could be used to increase the likelihood of suggesting the more correct word.

Though they were the first to use the Web to improve spell-checking, other researchers had previously used the Web as a corpus. Zhu and Rosenfeld compared 3-grams, probability estimates of three

---

<sup>12</sup> <http://developer.apple.com/documentation/Cocoa/Conceptual/SpellCheck/SpellCheck.html>

consecutive words, to model natural languages[2]. They compared results obtained from the Web and the British National Corpus<sup>13</sup>, and had more success using the Web. The EU MEANING project uses the Web as a data source for word sense disambiguation, finding the correct meaning of a word with multiple definitions.

The team used several metrics to detect misspelled words and suggest corrections. They utilized several features that used the Web and took context into account, such as the number of search results for the word, and the probability of bi- and tri-grams (the amount of results for the word and the surrounding words.) They also used metrics that did not require the web, such as keyboard and edit distance.

The results of their research were that the SIF percentage of their WebSpell program was higher than Word, Aspell and Ispell. According to Jacquemont, they saw that “Web-based features have a significant impact on the efficiency of the system.” This was their hypothesis and it was confirmed by their experiment. Therefore, if we somehow model their experiment we can expect the same results.

## 6 Procedures

Like the most spell checkers, our program scans a document and finds words that are not in its lexicon (in this case, a computer file containing a word list for the English language). The problem that a spell-checking program solves is one of picking from an arbitrarily large lexicon the single word that best corrects an unknown token in a document. We believe that the easiest way to solve this problem is to program several small tasks. Each task is responsible for one metric, and each metric makes the spell checker more accurate. We then combine the metrics using proportionality constants

---

<sup>13</sup> <http://www.natcorp.ox.ac.uk>

that we determined subjectively based on how effective each metric is. Given enough time to develop and run a sufficiently sophisticated system, we could use the methods of calculus to determine the optimal values for these constants. After the metrics are combined, each suggestion has a score and the word with highest score is suggestion most likely to be correct. We decided to create metrics that the spell checker can use to create a list of possible corrections and determine which token from that list is most likely to be correct. During the project, we used five metrics: three non-internet metrics and two internet metrics.

### 6.1 The Corpus

One of our metrics was to compare the misspelled word not only with a Web corpus but also with corpora of our own. The purpose is to compare whether you receive a more accurate result with a Web corpus or corpora that we have chosen. A corpus is “a collection of texts which have been selected and aggregated so that language can be studied on the computer”<sup>14</sup>. A corpus is also used for “statistical analysis, checking occurrences or validating linguistic rules...”<sup>15</sup>.

A corpus should have certain characteristics: it should be a sample of something, have a finite size, be machine readable, and be a standard reference[2]. Also, it should be representative of the subject it covers, including a broad range of topics with reasonable depth[6].

We will create our own corpora in English and Hungarian from documents freely available through the Project Gutenberg website. We used a variety of documents in order to create a collection that would be representative of each language, including novels as well as

---

<sup>14</sup> <http://www.ahds.ac.uk/creating/guides/linguistic-corpora/preface.htm>

<sup>15</sup> <http://en.wikipedia.org/wiki/Corpus>

scientific works, and using material from different time periods. However, we will use mostly material from this century, in order to create a model of the modern language.

We will use our corpora to generate probabilities of letter combinations (our  $n$ -graphs metric.)

This will create a list of “illegal digraphs” that never appear in the document. The program will use these to narrow down the list of suggested replacements for the misspelled word, since suggestions that contain these digraphs cannot be words.

## 6.2 The Metrics

### 6.2.1 Edit Distance

The first of these metrics is *Damerau-Levenshtein distance*. Damerau-Levenshtein distance, also called *edit distance*, describes how many transformations a word must undergo to become another word. For example, the word “qwer” is 2-edit distances from “qwety”. There are four possible transformations: addition of a letter, deletion of a letter, transposition of two adjacent letters, and substitution of one letter. Using only 2-edit distances easily generate a long list of possible corrections, so we are only using 1-edit distance. Each 1-edit distance word is checked against our lexicon, and those that are in our lexicon become a suggestion.

### 6.2.2 Google Hits

The number of Web-search results is next metric. We put each possible correction into the Google search engine and check how many hits each word gets. Since more commonly used words will obviously get more hits, this metric is less heavily weighted than others. More emphasis is placed on the next metrics.

### 6.2.3 Letter $n$ -graphs

Checking letter  $n$ -graphs in a word is another metric.  $n$ -graphs are a permutation

of objects of length  $n$ , with the objects in this case being letters. Using  $n$ -graphs makes the spell checker more language specific by focusing in on *phonemes*, the smallest distinct units of sound used in human speech. We use our corpora to collect  $n$ -graphs into a database.  $n$ -graphs help our spell checker find misspellings more easily by questioning  $n$ -graphs that appear very little or never at all in the target language. Also, words that use more common  $n$ -graphs are more likely to be correct because those words use more common phonemes. After this metric is applied, the number returned is proportional to the number of  $n$ -graphs each word contains. Since we lack dictionaries that contain proper nouns and slang, this metric will help us eliminate possible suggestions by removing words that have  $n$ -graphs that appeared very little or not at all in our corpora for that language.

### 6.2.4 Keyboard Distance

The next metric we used is keyboard distance. Keyboard distance is determined by how far apart letters of the misspelled word are from those of the correction based on the location of the constituent letters of each word on the keyboard. This metric is very effective in finding a good correction since a common typing error is hitting a key adjacent to that which was intended to be struck. Small keyboard distances contribute to higher final scores.

### 6.2.5 Word $n$ -graphs

Our next metric is word  $n$ -graphs, which – put simply – uses the context of the misspelled word to find the best correction. This metric determines the number of results on the Web of each of the possible corrections with the  $n$  word-long phrase surrounding in order to find the best replacement. Word  $n$ -graphs work similarly to the language model, a statistical way of showing the probability of words appearing in a specific sequence. When our spell

checker finds a misspelled word, it uses three specific  $n$ -graphs: the previous-suggestion bigraph, suggestion-post bigraph, and the previous-suggestion-post trigraph. This is the most effective metric because it deals with words in the way they appear in language.

### 6.2.6 Agglutinative Breakdown

Agglutinative breakdown is our final metric. This metric breaks apart words into their stems and affixes based on a database collected from the corpus. This helps with the Hungarian spell checker because Hungarian is completely agglutinative language, sometimes including more than three affixes. Breaking apart longer words also helps make the spell checker more effective by leaving correctly spelled stems alone. Agglutinative breakdown has not yet been included in our spell checker. Breaking apart longer words will make the spell checker more efficient. In Hungarian, for example, compound words can be many words combined into one word; therefore, when we break it apart, we can leave the correct stems alone and only focus on misspelled stems.

## 7 Extending Beyond English

After we complete our tests and our code structure for the English language, we will have to overhaul the code and convert it so that it may be suited for the Hungarian language tests.

### 7.1 Hungarian

Agglutinative languages, such as Hungarian, Finnish, and Basque have proved to be a challenge for spell checkers. Their name comes from the Latin verb “agglutinare”, which means “to glue together.” Words are formed through the joining of morphemes, the smallest meaningful units of language. In these

languages, changing the form of a word (for example, changing the tense or case) does not require a change in the root word, but a change in (or, more than likely, the addition of) one or more affixes. Many words have a high affix to morpheme ratio, and many contain several affixes. Each root word can have an enormous number of possible forms; each Hungarian verb can have up to 5,070, not all of which are included in dictionaries. This creates difficulties for spell-checking programs.

In order to make checking more efficient and accurate, our program breaks words down into their constituent stems and affixes. It does this by comparing each word to a list of all possible affixes in the language, then spell checking each stem and affix.

In addition, a Hungarian alphabet, corpus, and dictionary need to be implemented into the program. If the program spell-checks a Hungarian document accurately after these changes, this will provide evidence that our program is extensible to any language.

## 8 Tests

In the preliminary stages of testing our spell checker, we have to create a document that is misspelled so that it may be inputted into the checker for testing. To create the misspelled documents, we will create five paragraphs with no general main idea. We will then ask random students in our school to rewrite these paragraphs. The conditions will be that they cannot use any tool to rewrite a word that they believe is misspelled, which meant that they will not be able use the Backspace key on the keyboard. Also, we will read it as clear as possible but as fast as possible. We will also read out when the sentence ends and when a punctuation mark is needed. We will make it optional for the test subject to look at the

keyboard as well as to look at the monitor while he is typing. At this point we will be able to test our spell checker in English and be able to obtain results from our test.

## 9 Results

In a sample set of 47 misspelled words, our system correctly respelled 75% of the words. 17% of the time the system chose the incorrect possible correction for a misspelled word and 8% of the time the system was unable to generate any possible corrections.

Unfortunately, we did not have the same luck with our Hungarian version. One issue that we had for the Hungarian spell checker was that the dictionary for the language did not include certain words and it did not include certain pronouns that could have been corrected by our spell checker. There were other small problems that inhibited us from producing an adequate spell checker to run tests on.

## 10 Discussion & Future Work

There are a number of steps we can take to improve our spell checker. One situation that the corrector cannot handle is frontier words such as “pieceof” which would correct two “piece of.” Also, the spell checker cannot correct two back to back misspellings. Finally, the weightings of each metric need to be further tested to be sure they are fine tuned.

The other issues with our spell checker deal primarily with optimization. Spell checking a word is currently a slow process. The two steps that consume the most time are generating possibilities for a misspelled word and querying Google for result counts. Both steps can be optimized through caching. Caching is a technique that would save possible misspellings for later use so they do not need to be generated on the fly.

Caching the possible misspellings would allow us to extend our possibilities from one edit distance to two edit distances and allow our spell checker to be even more accurate. Google results can also be cached, but for a limited amount of time. Google is a dynamic corpus and the result counts would need to be refreshed periodically.

Lastly, we would like to port our spell checker to more and more languages. Our code makes completing this task fairly trivial, and can be achieved with only a basic word list and corpus.

## 11 Conclusion

After evaluating our spell checker’s performance it is quite clear that utilizing Google is extremely helpful in ranking the suggestions. The result count returned from querying the possibility and the words surrounding it almost always returned the correct result.

## Acknowledgments

**Blase Ur**, Program Coordinator: Governor’s School of Engineering and Technology  
**Marc L’Heureux**, GSET Alumni and Linguistics Student at Boston College  
**Wanda Durán**, RTA Mentor  
**Governor’s School**, Dean Donald Brown  
**Governor’s School Board of Overseers**  
**Sponsors:** Prudential, Morgan Stanley, Rutgers University, The John and Margaret Post Foundation, John and Laura Overdeck

## References

[1] Karlsson, F. (1996, May 13). *The word-forms of the Finnish noun kauppa 'shop' (N=2,253), generated automatically*. Retrieved July 14, 2008, from <http://www.ling.helsinki.fi/~fkarlssso/genkau>

[2.html](#).

[2] S. Jacquemont, F. Jacquemont and M. Sebban. *Correct your text with Google*, 2007.

[3] Spell Checker – Wikipedia, the Free Encyclopedia. Retrieved July 10, 2008, from [http://en.wikipedia.org/wiki/Spell\\_Checker](http://en.wikipedia.org/wiki/Spell_Checker).

[4] *Hungarian Language* – Wikipedia, the Free Encyclopedia. Retrieved July 10, 2008, from [http://en.wikipedia.org/wiki/Hungarian\\_Language](http://en.wikipedia.org/wiki/Hungarian_Language).

[5] Spell Checker – Wikipedia, the Free Encyclopedia. Retrieved July 10, 2008, from [http://en.wikipedia.org/wiki/Spell\\_Checker](http://en.wikipedia.org/wiki/Spell_Checker).

[6] A. Kilgarriff, G. Grefenstette. Introduction to the Special Issue on the Web as Corpus, 2003.

[7] *Agglutinative language* - Wikipedia, the Free Encyclopedia. Retrieved July 17, 2008, from [http://en.wikipedia.org/wiki/Agglutinative\\_language](http://en.wikipedia.org/wiki/Agglutinative_language)

[8] English Language – Wikipedia, the Free Encyclopedia. Retrieved July 17, 2008, from [http://en.wikipedia.org/wiki/English\\_language](http://en.wikipedia.org/wiki/English_language)