Spelling Correction in Agglutinative Languages

Kemal Oflazer and **Cemaleddin Güzey**
Department of Computer Engineering and Information Science
Bilkent University
Ankara, 06533, Turkey
ko@cs.bilkent.edu.tr

1 Introduction

Spelling correction is an important component of any system for processing text. Agglutinative languages such as Turkish or Finnish, differ from languages like English in the way lexical forms are generated. Typical nominal or a verbal root may generate thousands (or even millions) of valid forms which never appear in the dictionary. For instance, we can give the following (rather exaggerated) example from Turkish:

uygarlaştıramayabileceklerimizdenmişsinizcesine[1]

whose morpheme breakdown is:

uygar	*-laş*	*-tır*	*-ama*
civilized	+BECOME	CAUS	+NEG
-yabil	*-ecek*	*-ler*	*-imiz*
+POT	+FUT	+3PL	+POSS-1SG
-den	*-miş*	*-siniz*	*-cesine*
+ABL	+NARR	+2PL	+AS-IF

Methods developed for spelling correction for languages like English (see the review by Kukich (Kukich, 1992)) are not readily applicable to agglutinative languages. This poster presents an approach to spelling correction in agglutinative languages that is based on two-level morphology and a dynamic-programming based search algorithm. After an overview of our approach, we present results from experiments with spelling correction in Turkish.

2 Spelling correction algorithm

Our approach comprises two-steps: (1) determining all the roots from the dictionary that can be the root of the misspelled word, and (2) generating (systematically) all the possible words that "resemble" the given character string, from these roots. The first step of the problem is relatively easy because of the static structure of the root dictionary. Techniques developed for spelling correction, say, in English can usually be applied here. The second step involves producing all the possible words from the selected roots and requires a generate and test search procedure.

We denote the set of the surface forms of the roots in the language by R. We denote by X and Y strings formed using the alphabet of the language. X will denote the surface form of the incorrect or misspelled string, and Y will typically denote the surface string that is a (possibly partial) candidate word. Y_{lex} will denote the lexical form of this candidate string.[2] We assume the existence of a function, $surface()$ to generate surface strings from lexical strings, i.e., $surface(Y_{lex}) = Y$. The edit distance metric $ed(X, Y)$ (Du and Chang, 1992) measures how many unit operations (*insertion, deletion, replacement* of single character and *transposition* of two adjacent characters) are necessary to convert one string X into another Y.

We would like to abstract the behavior of a morphological generator and analyzer for the given language by two finite state automata. A finite state generator $M_g = (P, \delta, V, S, F)$ where P is a set of states, δ is the state transition function, V is the output alphabet, S is the starting state, and F is a set of final states, generates, all correctly formed words of the language. The transitions of M_g are of the form $\delta(p_i) = p_j$ (p_i and $p_j \in P$), with an output $v_k \in V$ which denotes the lexical form of a morpheme in the language and also labels the transition. It should be noted that it is possible to go from one state p_i to another p_j by more than one transition, outputting a different morpheme. We say a string Y_{lex} is generated by M_g, if Y_{lex} is formed by concatenating, in order, the outputs of the machine as we traverse starting from S to one of the states in F. We denote by $L(M_g)$ as the set of all lexical strings generated by M_g. We also assume a finite state recognizer M_r which recognizes whether a given surface string is in the language or not.

The spelling correction problem can now be formulated as follows: Given an incorrect word X (rejected by M_r), and an edit distance threshold t, find the solution set of possible correct words $S(X, t) = \{Y | ed(X, Y) \leq t$ and $Y = surface(Y_{lex})$ and $Y_{lex} \in L(M_g)\}$ in viable time and space.

[1] This is an adverb meaning roughly "(behaving) as if you were one of those whom we might not be able civilize."

[2] Lexical and surface are used in the two-level morphology sense.

The set of all the possible roots for the incorrect word X is defined as $PR(X,t) = \{r \mid ed(X[i],r) \leq t$ and $1 \leq i \leq m$ and $r \in R\}$.[3] We assume that $PR(X,t)$ can be computed by a standard $q-$gram technique. Using a small number (3 - 5) of 2-grams, gives satisfactory solutions in our case.

Assuming that we have a set of root words found as described above, we now have to generate words in the language having these roots, that do not deviate from the given misspelled string by more than the threshold. The solution requires a generate and test probing of the finite-state automaton M_g, starting with the start state S. We now have to find all the paths from this state to one of the final states using the roots in $PR(X,t)$, so that when the morphemes along this path are concatenated and surface string is generated, it is within an edit distance t of X.

When the search starts morphemes are concatenated to root and the length of the candidate lexical string Y_{lex} increases. After one step of the search, the partial surface string Y is compared with a suitable prefix of X. In most of the cases the candidate Y will deviate from these prefixes of X by more than the threshold without reaching a final state, so that we can no longer get to a viable solution. In such cases we do not consider any further transitions from that state. Special attention has to paid when a suffixation changes the surface realization of morpheme immediately to the left.

3 Results from experiments with spelling correction in Turkish

We first present a spelling correction example from our implementation where we used bigrams ($q = 2$), and we chose the number of bigrams to use for determining the root, k, as 3 (see Figure 1). We tested

EXAMPLE

Misspelled word:	çaışmalarıyla	
Threshold t:	2	
Solutions:	yazışmalarıyla	yatışmalarıyla
on left edge	yapışmalarıyla	yakışmalarıyla
	...	...
	kaşışmalarıyla	çıkışmalarıyla
Candidate Roots:[4]	çağ çakı çal çalı çam çan çap çar çat çatı çağ çak çakış çal çalış çap çat çatış çav çav çay	

Solutions:[5]	Lexical	Surface
Edit distance 1	çat+Hş+mA+lArH+ylA	çatışmalarıyla
	çap+Hş+mA+lArH+ylA	çapışmalarıyla
	çalış+mA+lArH+ylA	çalışmalarıyla (corr.)
Edit Distance 2	çav+mA+lArH+ylA	çavmalarıyla
	...	...
	çat+mA+lArH+ylA	çatmalarıyla

Figure 1: Spelling correction example

[3] $X[i]$ denotes the string of the first i characters of X.

[4] The duplicate entries in the list of candidate roots in fact have different part-of-speech categories andhence different morphotactics.

[5] A small subset of the whole solution set is given, due to space limitations.

Table 1: Average number of operations per misspelled word.

Ed.Dis.	Rec.	Gen.	Ed. Dis. Oper	Soln.	% Accuracy
1	30.9	311.2	2498.4	3.6	95.1
2	108.4	4462.0	20680.4	52.0	95.1

our algorithm on a set of 141 randomly selected incorrect words from Turkish text with edit distances 1 (86%) and 2 (14%) to their correct form. The morphological analyzer and generator that we used was our two-level specification for Turkish (Oflazer, 1993), developed using the PC-KIMMO system. It is, however, rather slow and can analyze only about 2 forms per second and can generate about 50 forms a second on Sun Sparcstations. So, instead of using timings, we counted the number of times certain expensive operations we called. The statistics shown in Table 1 show the average number of morphological recognitions and generations, and the edit distance operations required, and the number of correct solutions offered *per misspelled input word.* The last column indicates the percentage of cases the intended correct form was found. We also have developed a algorithm for ranking the solutions which offers the intended correct form as the first in 74% of the cases when $t = 1$.

4 Conclusions

This poster has presented a spelling correction algorithm for agglutinative languages that is based on a two-level morphological generator and analyzer, and a intelligent generate and test search procedure.

5 Acknowledgement

The research was supported in part by a NATO Science for Stability Grant, TU-LANGUAGE.

References

M .W. Du and S. C. Chang. 1992 A model and a fast algorithm for multiple errors spelling correction. *Acta Informatica*, 29:281–302.

K. Kukich. 1992 Techniques for automatically correcting words in text. *ACM Computing Surveys*, 24:377–439.

K. Oflazer. 1993 Two-level description of Turkish morphology. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, April. A full version appear in *Literary and Linguistic Computing*, Vol.9 No.2, 1994.