Turkish Journal of Electrical Engineering and Computer Sciences

Volume 26 | Number 5 Article 5

1-1-2018

Large vocabulary recognition for online Turkish handwriting with sublexical units

ESMA FATIMA BİLGİN TAŞDEMİR

AYŞE BERRİN YANIKOĞLU YEŞİLYURT

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 2218 – 2233

doi:10.3906/elk-1801-234

Large vocabulary recognition for online Turkish handwriting with sublexical units

Esma Fatıma BİLGİN, Ayşe Berrin YANIKOĞLU YEŞİLYURT*
Computer Science and Engineering Program, Faculty of Engineering and Natural Sciences, Sabancı University, İstanbul, Turkey

Received: 25.01.2018 • **Accepted/Published Online:** 20.04.2018 • **Final Version:** 28.09.2018

Abstract: We present a system for large vocabulary recognition of online Turkish handwriting, using hidden Markov models. While using a traditional approach for the recognizer, we have identified and developed solutions for the main problems specific to Turkish handwriting recognition. First, since large amounts of Turkish handwriting samples are not available, the system is trained and optimized using the large UNIPEN dataset of English handwriting, before extending it to Turkish using a small Turkish dataset. The delayed strokes, which pose a significant source of variation in writing order due to the large number of diacritical marks in Turkish, are removed during preprocessing. Finally, as a solution to the high out-of-vocabulary rates encountered when using a fixed size lexicon in general purpose recognition, a lexicon is constructed from sublexical units (stems and endings) learned from a large Turkish corpus. A statistical bigram language model learned from the same corpus is also applied during the decoding process.

The system obtains a 91.7% word recognition rate when tested on a small Turkish handwritten word dataset using a medium sized (1950 words) lexicon corresponding to the vocabulary of the test set and 63.8% using a large, general purpose lexicon (130,000 words). However, with the proposed stem+ending lexicon (12,500 words) and bigram language model with lattice expansion, a 67.9% word recognition accuracy is obtained, surpassing the results obtained with the general purpose lexicon while using a much smaller one.

Key words: Online handwriting recognition, Turkish handwriting recognition, hidden Markov models, statistical language modeling, UNIPEN, grammatical sublexical units, delayed strokes

1. Introduction

Online handwriting recognition is the task of interpreting handwritten input, at character, word, or text line level. The handwriting is represented in the form of a time series of pen tip coordinates that are captured by digitizer equipment, such as tablets and pen-enabled smart phones. Writing style can be either constrained to be hand-print only or fully cursive, or totally unconstrained.

In handwriting recognition, language modeling is used to improve the results of the optical module. Typically, a lexicon containing the vocabulary for a given task is used to restrict the word hypotheses. One such example is the roughly hundred-word lexicon of numbers used for bank check processing. In contrast, often a 30,000-word lexicon is used for general purpose English text recognition systems.

Turkish is an agglutinative language where new words are formed by adding suffixes to the end of root words. There are grammatical rules (i.e. *morphotactics*) governing which suffixes may follow which other, and in what order, but the number of possible words that may be generated by adding suffixes is practically infinite, with application of some suffixes repeatedly. As such, a finite-size lexicon for Turkish would miss a significant

*Correspondence: berrin@sabanciuniv.edu

percentage of Turkish words, causing a high out-of-vocabulary (OOV) rate. This makes lexicon-based large vocabulary text recognition approaches unsuitable for Turkish, or other agglutinative languages.

The existence of many characters with diacritics exacerbates the so-called *delayed stroke* problem in online handwriting recognition. When a stroke is separated from the character body it belongs to by one or more strokes, it is said to be 'delayed'. For instance, the dot of an 'i' or the cross of a 't' can be considered as delayed when they are not written immediately after the corresponding letter body. Writers have different writing practices regarding delayed strokes that cause variations in the writing order. These variations pose a problem for systems that model the writing as a sequence, and consequently affect the recognition performance negatively.

This study presents a large vocabulary recognition system for online Turkish handwriting, using hidden Markov models (HMMs). The system obtains state-of-art recognition rates (comparable to those for English) by: i) leveraging the available dataset for English, to construct a recognizer for Turkish for which the available data are scarce; ii) addressing the delayed strokes problem by detecting and removing them during preprocessing; iii) addressing the high out-of-vocabulary rates observed with fixed size lexicons by using sublexical units (stems and endings) together with a bigram language model.

2. Related work

There have been many studies since the early 1990s on the online handwriting recognition problem [1]. Initially limited to the recognition of isolated characters and digits, the state-of-the-art research now takes aim at unconstrained word and sentence recognition. While much of this research is focused on the recognition of Latin-based alphabets, and especially English, handwriting recognition in other scripts has also been gaining attention in recent years [2–4].

Different techniques and approaches are used in recognition systems. HMMs have been the most popular technique for first offline handwriting recognition [5, 6] and then online handwriting recognition for a long time [7–10]. HMMs can be used for modeling strokes, characters, words, or sentences. Once the HMM model is defined with a given number of states, a topology, and emission distribution, the parameters of the models (transition and emission probabilities) are then learned by means of the expectation maximization algorithm. Decoding, on the other hand, is done through the Viterbi algorithm that finds the most likely path within the HMM. We do not include the details of HMMs here, but an excellent tutorial can be found in [11].

More recently, the deep learning techniques have surpassed HMMs in both popularity and performance, especially in problems where a large amount of training data is available [12, 13]. In particular, long short-term memory neural networks (LSTMs) have been very successful in both online and offline handwritten and machine-print text recognition problems in recent years [14]. Despite the success of deep learning systems, HMMs remain a viable alternative for sequence learning tasks, especially in cases of limited data and computational resources. HMMs are also used in hybrid systems together with different kinds of artificial neural networks (ANNs), including deep learning methods such as recurrent neural networks (RNNs) and LSTMs [15–23]. For instance, ANNs are employed for extending the HMM with contextual information [15, 16], feature extraction [17, 20], and predicting emission probability densities in an HMM system [19, 21].

There is little research about recognition of handwritten Turkish text. A number of studies cover offline Turkish character recognition with some constraints applied on the style or the case of writing [24–26]. In offline handwritten Turkish text recognition, Yanikoglu and Kholmatov [27] use the HMM letter models previously

developed for English, by mapping the Turkish characters to the closest English character (the input of the word güneş is recognized as gunes). They evaluate the use of a Turkish prefix parser to detect non-Turkish word prefixes during decoding, as an alternative to using a lexicon. Authors report a 56% top-10 word recognition rate using a 17,000-word lexicon obtained from a newspaper corpus and around 40% using the Turkish prefix parser. Resembling the work in [6], [28] uses a character-based word recognition method for offline lowercase mixed-style handwritten Turkish words and achieves an 84% recognition rate using a lexicon of size 2500.

In online handwriting recognition, Vural et al. [29] present a comprehensive evaluation of various HMM architectures and parameters and report a 94% word recognition rate using a 1000-word lexicon using character HMMs. They report that about 35% of the errors are due to the removal of diacritical marks in preprocessing.

3. Challenges in Turkish handwriting recognition

3.1. Delayed strokes

The modern Turkish alphabet, adopted in 1928, is based on the Latin alphabet. It differs from the English alphabet by the addition of five characters with diacritics (ç, ğ, ö, ş, ü) and the diacritic-free version of 'i' (ı), with omission of three others (q, w, and x). Diacritical marks or other integral parts of characters like the cross-stroke of 't' can be written in delayed fashion, which causes a challenge in online handwriting recognition by introducing an extra source of variation in the sequence order of the handwritten input.

The delayed strokes must be detected before any handling is applied. Exact delayed stroke detection can only be done after recognition, or more specifically after letter boundaries are known, by considering those letter parts that are written separately from the corresponding character bodies. For instance, the dot of an i is not considered delayed if it is written right after the letter body, even though it involves a pen-up movement with a backward move of the pen. Nonetheless, there have been various approximate definitions, such as calling all backward moves after pen-up as delayed strokes, so as to detect and handle delayed strokes during preprocessing.

Once a procedural definition is at hand, the delayed strokes can be detected and then handled according to a chosen method, of which there are a few. One simple alternative suggests to remove the delayed strokes, assuming that they are redundant in recognition [30, 31]. However, delayed strokes are useful for resolving confusions between similar words in some scripts like Arabic and Farsi. Another option is to *reorder* the writing sequence by replacing delayed strokes in their relevant positions. Some handcrafted rules [32], vertical projection [33], and oversegmentation [34] are the methods used for deciding attachment points of delayed strokes. In the *hat-feature* representation, delayed strokes are removed and remaining data points are marked with a binary feature [10, 14, 15, 34]. The feature takes on the value of one at locations that was under a removed delayed stroke, and zero otherwise. Exact delayed stroke modeling represents delayed strokes as special characters in the alphabet [7, 29] by adding all alternative spellings of each word with delayed stroke(s) to the lexicon. This approach has the disadvantage that the lexicon size can grow significantly. A final alternative suggests to move the delay strokes at the end of the word, in order to normalize the writing order [35].

Based on our extensive research on delayed strokes and the handling methods for them, we opted for removal of delayed strokes once they are detected according to our definition. More details of our approach are given in Section 4.1.

3.2. High out-of-vocabulary rate

Language modeling is an essential part of all modern recognition systems, used for improving the recognition results by imposing some constraints on the decoding procedure. Building a suitable lexicon (recognition

vocabulary) and integrating the linguistic knowledge (i.e. occurrence statistics of vocabulary items) into the decoding process are two main tasks in language modeling. During the HMM decoding process, the most probable path is searched through a probabilistically scored time/state lattice (network). In this process, a list of words (referred as the lexicon) is used for limiting the search of probable paths to the valid words. The size of the lexicon is thousands of words for larger vocabulary recognition systems.

Turkish, which is an agglutinative language, has a highly productive inflectional and derivational morphology that leads to a theoretically infinite lexicon and consequently high out-of-vocabulary (OOV) rates even with large lexicons. The term OOV refers to the words that appear in the testing set but not in the lexicon. OOV words translate to errors since they cannot be recognized by the system, which is constrained to match the input to the vocabulary items in the lexicon. In order to deal with the high growth rate of vocabulary in languages such as Turkish, it has been proposed to use lexicons based on smaller units. The sublexical units like stems, morphemes, and morphs (statistically derived subwords) can be obtained by splitting words grammatically or statistically. Once the lexicon is designed with sublexical units, the text corpus is adapted to sublexical units and words are represented in terms of these units. In the next stage, the n-gram language models are trained over these morpholexical units.

This approach is recently applied for language modeling in the automatic speech recognition (ASR) domain [36–39]. In [36], units like stem+endings, stem+morphemes, and syllables and their combinations are investigated instead of words. Similarly, Erdoğan et al. propose a combined sublexical units system within the weighted finite state transducer (WFST) framework. In [39] and [38], Sak et al. integrate morphology into a WFST and experiment with lexical morphemes and statistical morphs. In [40], statistical morphs are used for language modeling.

Similar to the approaches taken in Turkish ASR, we propose to build a lexicon based on stem+ending sublexical units and use it in the decoding step as a solution to the OOV problem in Turkish large vocabulary handwriting recognition. The details of our approach is given in Section 5.

3.3. High level of confusability

Another problem of Turkish related to its agglutinative nature is the high level of confusability between word surface forms, which increases with the vocabulary size. As words are created by affixation to root words, many words share the same stem and often differ only in suffixes by a single character. The problem is more severe when these suffixes have similar optical features, as in the example of *kitabımı* (my book - accusative) and *kitabını* (your/his/her book - accusative). In this case, the recognizer would have to chose between two alternatives with almost equal optical and linguistics scores. Even with 0% OOV rate, it is still difficult to make a correct recognition because of the high confusability in a large lexicon. While there is no solution to this problem, it is worth mentioning among problems pertaining to the recognition of Turkish text.

4. Proposed recognition system

4.1. Preprocessing

In order to eliminate variations in data, we use a series of preprocessing methods, some of which can be found in the literature. A solution for delayed strokes is applied at this stage as well. Specifically, we apply size normalization in the form of rescaling the height of the writing to 1000 pixels, while the width is scaled accordingly, preserving the aspect ratio. Skew normalization is simply done by correcting the baseline angle and

it is found to be useful in increasing the recognition accuracy. For slant normalization, the histogram method in [41] is preferred.

We also apply equidistant resampling by linear interpolation on pen trajectory. The number of sample points is set proportional to the number of characters within the word. Note that the number of characters is known a priori for the training set, from the known labels of the input words. As for the test set, we used a heuristic method for estimating the number of characters in the input word as described in [12]. With this method, the number of lines crossing the midline (horizontal line between the baseline and corpus line) is used to estimate the number of characters in the word.

We remove the delayed strokes using a decision tree classifier developed in this work. We trained the classifier to detect delayed strokes, using strokes found in 1000 randomly selected UNIPEN words. The decision tree learns to identify delayed strokes automatically from the given training data, based on their characteristics as size, relative position, and overlap with the rest of the word. It essentially identifies as delayed those strokes that start after a pen-up and a backward move if they also have somewhat small sizes/overlaps and appear at higher/lower parts of the word, with an accuracy of 10.3%. We have observed a 2.03% word accuracy improvement when delayed strokes are removed, compared to leaving them in the input.

4.2. Feature extraction

Different handcrafted features are used in the literature for the representation of online handwriting. Starting from a wider set of features that are commonly used [7, 12–15, 34, 35], we have selected a subset of 8 features through extensive experiments, according to the word recognition performance of the system. Then, considering the errors made by the system, a new feature (distance to median y-value) is designed within this work. The new feature adds some global context information to the feature vector, by indicating the vertical position of the frame with respect to the median y-values in the whole sequence. This information is relatively robust if there is not a heavy slope on the baseline and helped improve the overall recognition accuracy. The nine features are:

- *delta*: differences from the x- and y-coordinates of the previous point;
- *sine and cosine of angle* between x-axis and the line joining consecutive points;
- *curvature angle*: the angle between the lines to the previous and the next point;
- *vicinity linearity*: average squared distance of each point in the vicinity to the straight line from the first to the last vicinity point;
- *vicinity slope*: a pair of features such that cosine and sine of the angle of the straight line from the first to the last vicinity point;
- *pen-up/down*: a binary feature showing whether a sampling point is an up point (pen is lifted up here) or a down point (pen is touching the writing pad);
- *normalized x*: the x-position taken after high-pass filtering, i.e. after subtracting a moving average of 5 previous points' x-values from the real horizontal position.
- *distance to median y-value*: distance to the median y-value of the given sample point sequence.

4.3. Hidden Markov model classifier

In order to deal with the extremely large vocabulary size, we use character models rather than word HMMs. Once trained, the character models are concatenated to obtain word models. An HMM is built for each of the 52 characters (uppercase and lowercase) in the English alphabet, which includes 46 characters common to the

Turkish alphabet. With the addition of 12 more models for the 6 additional Turkish characters (ç, ğ, ı, ö, ş, and ü), a total of 64 character models are trained and used in recognition.

Each of the character models has 20 states in linear topology with self and next transitions without skips. The number of states in a model is decided empirically on the basis of performance obtained with our feature set on the UNIPEN dataset. In the emitting states, the observation probability distributions are estimated using Gaussian mixtures with 35 components. The number of Gaussians is decided using an iterative splitting method as suggested in [42], where at each iteration the Gaussian distribution with the highest weight is split until no further improvement is obtained on a validation set.

The well-known HTK software [43] is used for system implementation. The recognizer is trained with samples from both the UNIPEN and ElementaryTurkish datasets, which are introduced in Section 6.1. Since the number of training samples of ElementaryTurkish dataset is very small, we have used the UNIPEN training data and Turkish data together for training. Specifically, models of characters that are common for English and Turkish are trained with both UNIPEN and ElementaryTurkish samples, while the others are trained with only the relevant samples.

5. Language modeling

As discussed in Section 3.2, the use of sublexical units is an effective language modeling solution to the OOV problem, especially in the case of morphologically rich languages. These sublexical units can be obtained by splitting words grammatically or statistically. The grammatical approach to obtain sublexical units has the advantage of preventing grammatically invalid productions via language information. The usual grammatical sublexical units can be syllables, morphological units such as morphemes, or decompositions into stems and endings (grouping of suffixes). In the present work, we take the grammatical approach of using stem+ending sublexical units to create a lexicon for the large vocabulary recognition of Turkish handwriting.

5.1. Stems and endings as lexical units

Words can be decomposed into stems and endings by a morphological analyzer. We employ TRmorph [44], an open-source morphological analyzer for Turkish based on finite-state transducer technology. We derive stem+ending lists from the text corpora that are introduced in Section 5.3.

It is a known issue that morphological analysis yields multiple decompositions for some words. Sometimes, it is due to derivation of multiple stems from the same root word. For example, decomposition of the word *gözlükçü* (optician), whose root is *göz*(eye), yields three stems including the root *{göz, gözlük (eyeglasses), gözlükçü}*. In other cases, there are stem(s) and root(s) that share the same spelling. For instance, the word *adaya* may have one of these three roots: a verbal root *ada* (to devote) and two nominal roots *ada* (island) and *aday* (nominee).

In the present work, we prefer to keep all the different stems and endings in the vocabulary in the case of multiple decompositions. With this approach, the frequency of stems and endings is directly related to the frequency of the words they are extracted from. Each stem and ending of a frequent word become frequent as well. As such, some rarely used stems and endings may have their frequency of occurrence artificially increased. On the other hand, the coverage of the vocabulary increases, as all possible stems (and endings) are included in the lexicon, in contrast to selecting only one of the alternative decompositions via morphological disambiguation.

In Turkish, vowel harmony is applied during suffix affixation, such that vowels of suffixes change to comply with the vowel harmony rules. For example, for the vowel group H = {'i', 'ı', 'u', 'ü'}, the verbal suffix dH is

realized as -di in "gel+di", -dı in "al+dı", -dü in "gör+dü", and -du in "oku+du", depending on the last vowel of the stem they are attached to. One solution for affixation complying with vowel harmony is to group stems according to their last vowels and then specify which suffixes are attached to which group of stems as done in [37]. This method has the advantage of smaller search states, which can potentially improve recognition performance. Another option is to employ a postprocessing step for correcting vowel harmony discrepancies after recognition. This approach allows a somewhat more flexible decoding; this is also the case when only a chosen number of endings are included in the lexicon, instead of all possible formations. If one form of realization is absent from the vocabulary (for example -dü) while another one that is functionally the same but different in realization is included, the system can still generate the correct ending after postprocessing as in "gel+dü" → "gel+di".

5.2. N-Gram language models

We adopt the statistical language modeling method of N-grams to score probabilities of paths in the recognition network. N-gram language models are trained on corpora of texts that are large enough to make good estimations of word sequences of the language. In the present work, we use the BOUN Web Corpus to train a bigram model for the stem+ending vocabulary.

Given a sentence S as an ordered set of words $W = w_1, 1_2, \ldots, w_l$, its probability can be written as [45]

$$P(W) = P(w_1, 1_2, \ldots, w_l) = \prod_{i=1}^{l} P(w_i | w_{i-1}, \ldots, w_1). \tag{1}$$

An N-gram model uses history of only $N-1$ words to estimate the right-hand side of the equation above, and so an approximation of Eq. 1 is

$$P(W) \approx \prod_{i=1}^{l} P(w_i | w_{i-1}, \ldots, w_{i-N+1}). \tag{2}$$

We use bigram statistics of stems and endings as the language model in our large vocabulary recognition system. There are two methods of integration of language model scores into the recognition networks: lattice rescoring and lattice expansion. A lattice is a weighted directed acyclic graph used in decoding, with paths from the start state to a final state. Each path represents an alternative decoding hypothesis with weights composed of the corresponding optical model likelihood and the language model probability.

With lattice rescoring, multiple hypothesis are generated in the form of a lattice during the decoding process and then reranked using a higher order language model. A lattice is created for each test sample by running the decoder with the recognition network [45]. The number of alternative hypotheses determines the size of the search space of the generated lattice. Keeping a high number of alternatives increases the computational complexity, while a small number of alternatives may result in missing the path for the correct transcription.

Lattice expansion is based on modifying the recognition network by adding extra nodes and edges as duplications of existing ones, so that all possible N-grams are represented [46]. Since the lattice expansion method uses the whole network rather than an abridged one, it provides a complete search space at the cost of increased complexity. We have considered both methods and found lattice expansion to give better performance.

5.3. Language modeling corpus

We use the The BOUN Corpus to build the lexicons (word level or subword level) and to generate a bigram language model for the stem+ending lexicon. The BOUN Corpus is composed of four subcorpora that are collected from websites by means of a web crawler script [47]. Three of the subcorpora are derived from the websites of three major newspapers in Turkish and they are referred to as BOUN NewsCor. The other subcorpus is obtained from a sampling of Turkish websites and is named BOUN GenCor. Statistics about the BOUN Corpus can be found in Table 1. In the table, the tokens column represents the number of words or lexical units such as punctuation marks, whereas the types column represents the number of unique tokens.

Table 1. BOUN Corpus details. *M* is used as an abbreviation for million.

Corpus	Words	Tokens	Types
NewsCor	184M	212M	2.2M
Milliyet	59M	68M	1.1M
Ntvmsnbc	75M	86M	1.2M
Radikal	50M	58M	1.0M
GenCor	239M	279M	3.0M
BOUN Corpus	423M	491M	4.1M

The BOUN corpus is automatically generated from mostly web resources and so it requires some cleaning and formatting to be usable in our system. We preprocess the corpus for extracting grammatical words, stems, and endings. As a first step, we remove all entities containing nonalphabetic characters (punctuation, numerals, monetary signs, etc.) to obtain 408M words, while keeping the letter case information. Secondly, we analyze the words morphologically to extract stems and suffixes. A total of 335M words are successfully analyzed, leading to the extraction of at least one stem. Some 73M words that cannot be analyzed include proper names, foreign words, misspellings, and strings of characters that do not constitute grammatical words, and finally words erroneously concatenated together.

Table 2 summarizes the number of grammatical units extracted by morphological analysis. We observe that the average number of stems extracted per word is about 4.5. About 20% of the words are left undecomposed during analysis, as they are either proper names or are in the root form already. When these words are excluded, the average number of stems per word becomes 5.7.

Table 2. Lexical units obtained from the BOUN corpus.

Unit type	Unique	Total
words	1,578,553	334,758,204
stems	69,013	1,528,104,572
endings	138,226	1,146,201,689

5.4. Lexicon size determination

The lexicon has a direct effect on the recognition performance; therefore choosing the size and composition of the lexicon is very important. A useful criterion for choosing an appropriate lexicon size is the coverage and OOV rates of the lexicon on a particular word set. In this work, we aimed to obtain 95% coverage of the BOUN

Corpus with both word-based and stem+ending-based lexicons, in order to be able to compare their overall performances.

Table 3. Coverage rates of word-based large vocabulary task.

Top-n	Size	Corpus coverage	Test set coverage
3%	50,000	90.0%	62.6%
5%	80,000	93.2%	69.6%
7.5%	120,000	94.4%	72.3%
8.2%	130,000	**95.7%**	75.6%
9.5%	150,000	96.3%	77.6%
100%	1.57M	100.0%	87.5%

Table 4. Coverage rates for stem+ending-based large vocabulary task (the chosen lexicon setting is given in bold).

Unit	Top-n	Size	Corpus coverage	Test set coverage
stem	10%	7000	96.6%	82.0%
ending	5%	7000	97.9%	90.7%
stem+ending			95.3%	83.9%
stem	7%	5000	94.5%	76.7%
ending	5%	7000	97.9%	90.7%
stem+ending			94.0%	82.1%
stem	10%	7000	96.6%	82.0%
ending	4%	5500	97.24%	87.3%
stem+ending			**95.0**%	**83.6**%
stem	10%	7000	96.6%	82.0%
ending	3%	4100	96.1%	82.4%
stem+ending			94.6%	82.7%

For our word-based lexicon, we start with the most frequent 50,000 words of the BOUN Corpus and keep adding words according to their frequency until we obtain 95% coverage. In Table 3, we give the corpus coverage rates for different size lexicons derived from the BOUN Corpus and the corresponding test set coverage. As can be seen in the table, a lexicon containing the most frequent 130,000 words has 95.7% coverage on the BOUN Corpus and 75.6% on the test set. It is worth noting that even the whole corpus only covers 87.5% of the test vocabulary.

As for the stem+ending lexicon, we follow a similar procedure to find a suitable vocabulary size satisfying a coverage of 95%. However, measuring coverage of a stem+ending lexicon is not a straightforward task, since the combination of stems and endings determines the overall coverage of the lexicon. If a stem is missing from a stem+ending vocabulary, then any word that has that particular stem will be missing from the vocabulary as well. Similarly, a missing ending translates to recognition errors for all words containing that ending.

We include all stems and endings resulting from the morphological analysis of a word in our stem and ending lists. When a word has multiple decompositions, we accept a word as covered if it has at least one decomposition that is covered by the lexicon. Continuing with the *gözlükçü* (optician) example given in Section

5.1, we obtain two stem+ending pairs: {göz+lükçü, gözlük+çü} and one stem with no ending, i.e {gözlükçü}, with the morphological analysis. It is sufficient for the lexicon to contain one of these three decompositions, to have the word gözlükçü covered by the vocabulary.

We choose the stems and endings according to their frequency of occurrence in the corpus. For a given lexicon, the coverage rate is calculated by generating all possible combinations of stems and endings (including empty endings). We observe that increasing the number of stems increases the coverage more compared to increasing the number of endings (see Table 4). We chose the top 10% of stems and 4% of endings according to frequency of occurrence to achieve the target coverage rate (95%) with minimum vocabulary size in our stem+ending design.

6. Experiments

In this section, we give details of the experimental setting for evaluation of different lexicon designs with associated decoding networks in the large vocabulary handwriting recognition task. With the configurations given in Section 4.3, we first train the system with UNIPEN and ElementaryTurkish datasets together. Trained character models are concatenated to form the word models.

We use two alternative representations for the lexicons: words and stems–endings with different vocabulary items. The sources of lexicons are either i) the ElementaryTurkish (ET) dataset lexicon, which contains 1950 words; or ii) the test set of the ET dataset analyzed into 873 stems and 495 endings; or iii) the BOUN Corpus, from which 130,000 words or 7000 stems and 5500 endings are selected as detailed in Section 5.4. Note that since the ElementaryTurkish dataset used for testing the system is quite small (800 unique words), the ET training set lexicon (1950 unique words), which covers the test set lexicon, is used in the decoding process for more trustable results. The effect of a bigram model is evaluated within the stem+ending solution. Results are reported separately for each setting.

We give three types of results: 1) the raw recognition results, which reflect the output of the HMM classifier; 2) results after a language model is applied; 3) results after applying a postprocessing step. The postprocessing is applied to the recognition results for resolving discrepancies between stem and suffix(es) regarding vowel harmony. This step is required since no constraints are imposed during affixation in the recognition networks. Moreover, in some cases consonant harmony is fixed by changing the last character of stems according to Turkish orthographic rules.

Language models are applied as lattice rescoring and lattice expansion to the recognition network. Word insertion penalties and grammar scale factors are optimized empirically on a separate validation set. SRILM [48], the well-known and widely used toolkit for building and applying statistical language models, is used for N-gram language model generation and its integration to recognition networks. The HVite tool of the HTK is used for recognition of test samples.

6.1. Handwriting datasets

A large collection of handwriting samples are required for training a recognizer for state-of-the-art performance, so that it can represent most of the variations in handwriting (e.g., cursive/handprint styles, different letter shapes, and slant). We use two such datasets to train our HMM model. In this section we give detailed information on the datasets and the way they are used for training the recognizer.

6.1.1. UNIPEN

We use the isolated word collection of the UNIPEN dataset for training character models common between English and Turkish. The UNIPEN online handwriting database is a collection of handwritten samples (containing sets of characters, digits, words, and texts), collected by a consortium of 40 companies and institutes over time. The whole collection consists of a total of 5 million characters by writers from all around the world, representing a wide variety of writing styles. The number of words contributed by different writers is highly unbalanced, ranging from less than 5 to more than 500. In fact, more than 2000 writers submitted very small numbers of words, specifically between 1 and 5, whereas there are 59 writers who contributed more than 400 words.

In this work we use the isolated word collection (category 6) of the current publicly available version called the *train_r*01*_v*07 training dataset. This collection contains 75,529 cursive or mixed-style words in total. The lexicon size is 13,913 words, with separate upper and lowercase versions for some of the words.

6.1.2. ElementaryTurkish handwriting database

As there was no public dataset for online/offline Turkish handwriting, previous studies from the literature used proprietary collections. We have collected around 10,000 samples of isolated words by means of an Android tablet. The dataset, which we refer to as the ElementaryTurkish (ET Dataset) dataset, contains words from the 2089-word lexicons of Turkish 1st and 2nd grade books. Volunteers, including children and people from different backgrounds (113 people in total), have contributed to the dataset by writing around 100 words per person on average. The subjects were instructed to write normally in an unconstrained manner. The dataset contains samples written in different writing styles (cursive, hand printed, and mixed style). Figure 1 shows a selection of samples from that dataset.

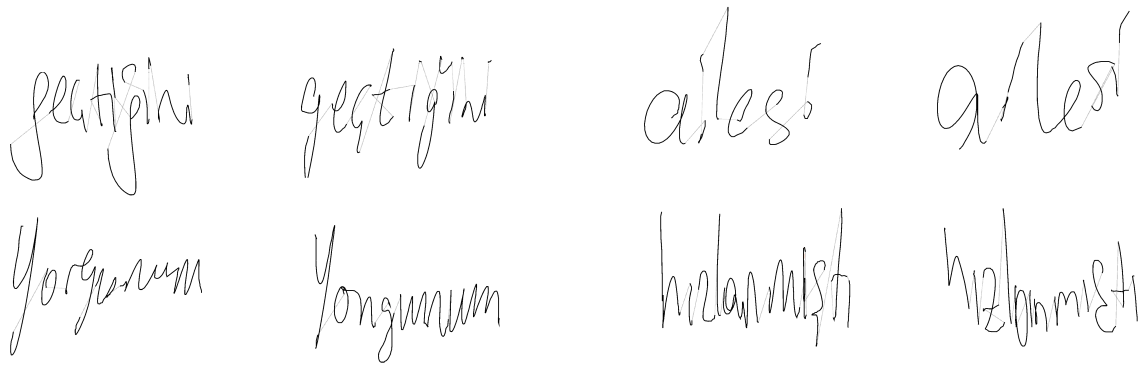

Figure 1. Sample handwritten words from the Elementary Turkish dataset: “geçtiğini”, “ailesi”, “Yorgunum”, “hızlanmıştı”.

We split the ET Dataset into three subsets as train, validation, and test sets in a writer-independent fashion (i.e. writers in each group are nonoverlapping), so that the test set gives an unbiased estimate of writer-independent accuracy. The train set includes 7360 samples from a 1950-word lexicon written by 79 writers and the test set contains 804 samples written by another group of 8 writers. The validation set consists of a separate portion of the dataset that is written in sentences as opposed to isolated words and consists of 1700 isolated word samples written by 26 writers.

6.2. Results

Table 5 shows the word recognition accuracies obtained with alternative lexicon types (word or sublexical units). While the main goal is to evaluate the performance with a general purpose lexicon, we also give results obtained with lexicons derived from the ElementaryTurkish dataset itself, to show the maximum accuracies possible.

Table 5. Word recognition accuracy on the test set, using different language models (the highest accuracies in bold).

Lexical units	Lexicon source	Lexicon size	Language model	Raw	Postprocessed
Words	ET Dataset	1950	-	**91.7%**	-
Stem+ending	ET Dataset	873+495	-	52.5%	63.0%
Stem+ending	ET Dataset	873+495	2-gram (expansion)	79.8%	**79.8%**
Stem+ending	ET Dataset	873+495	2-gram (rescoring)	62.7%	62.9%
Words	BOUN	130,000	-	**63.8%**	-
Stem+ending	BOUN	7000+5500	-	44.6%	49.0%
Stem+ending	BOUN	7000+5500	2-gram (expansion)	67.8%	**67.9%**
Stem+ending	BOUN	7000+5500	2-gram (rescoring)	62.9%	63.3%

Table 6. Examples from recognition errors.

Ground truth	Recognized
ağırdı	çağırdı
Kitabımı	Kitabın
baktık	taktik
Futbolcu	Futbola
kitapla	kitaptı

Ground truth	Recognized
Akşam	Aksam
Hayal	Hayat
ördeği	önceyi
dönüyorsa	donuyorsa
bulunan	bilinen

According to the results shown in Table 5, the highest accuracy (91.7%) is obtained when the word-based lexicon is derived from the test set itself, given here as the baseline. When the large, general purpose word-based lexicon derived from the BOUN Corpus is used, the accuracy drops significantly to 63.8%. The large drop in performance can be mainly attributed to the low test set coverage of that lexicon (75.6% as seen in Table 3), as OOV rates directly impact overall accuracies. Another factor affecting the performance is the higher confusability among words in a larger lexicon.

In contrast, the general purpose stem+ending lexicon extracted from the BOUN Corpus and used with the bigram language model applied via lattice expansion, results in the best accuracy of 67.9%. Note that despite a much smaller lexicon and having the same text corpus coverage, this accuracy is better than that obtained with a general purpose word-based lexicon (63.8%). With a stem+ending lexicon derived from the test set itself, the highest accuracy is 79.8%.

We observe that is important to use a language model with stem+ending lexicons, without which the accuracy falls significantly below word-based lexicon results (49.0% versus 63.8%). As for the methods for incorporating language models, lattice expansion performs better than lattice rescoring for the stem+ending lexicons with the bigram model. With the expanded lattice, higher recognition rates are achieved, probably due to the use of the full search space instead of the reduced lattice. Postprocessing is not very useful for stem+ending lexicons when N-gram models are used, but it improves the accuracy slightly when no language modeling is applied.

When we analyze the results of the best performing system, we see that 29% of 259 misrecognitions are due to single character errors (e.g., 'kaldırdığında' vs. 'kaldırdığımda', 'batmakta' vs. 'bakmaktan'). This highlights the high confusability problem discussed in Section 3.3. Furthermore, 6% of errors involve words that are similar when the diacritical marks are removed (e.g., 'çoktu' vs. 'çöktü', 'Akşam' vs. 'Aksam'), which is unavoidable with the chosen delayed stroke handling approach. The OOV errors are mainly due to stems missing from the lexicon: the OOV rate is 43% for stems (100 out of 233 unique stems) and 20% for endings (32 out of 158 unique endings). Almost all of the wrongly predicted words are grammatically correct, indicating the effectiveness of the bigram language model. Table 6 shows some examples from the misrecognized words along with their incorrect predictions.

Figure 2 shows the error distribution according to misrecognized word parts (stem, suffix, or both). As can be seen in the figure, the integration of language models has a clear positive effect on decreasing the errors in suffix parts in stem+ending vocabularies. However, it does not help with the stem errors; in fact, a slight increase is observed in the number of misrecognized stems.

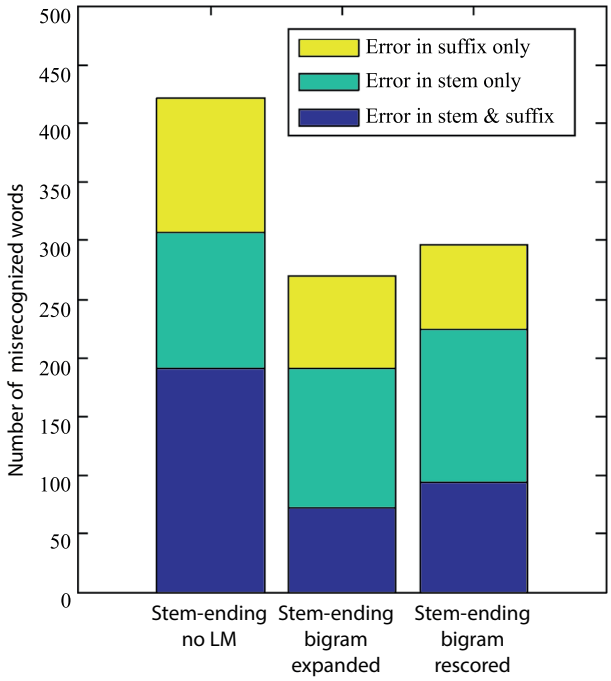

Figure 2. Distribution of errors for stem+ending vocabularies.

To the best of our knowledge, there is no other large vocabulary online Turkish handwriting recognition system in the literature. In online recognition with a small lexicon, Vural et al. obtain a 94% word recognition rate using a lexicon of 1000 words using character HMMs on a dataset of 3000 word samples written by 30 writers [29]. This accuracy is comparable to the 91.7% accuracy we obtained when using the 1950-word lexicon that covers the test set vocabulary.

7. Conclusion and future work

This work presents the first large vocabulary online handwriting recognition system for Turkish. The system achieves state-of-art results comparable to those obtained for English, through careful study of all components of

the recognition system, from preprocessing to language modeling. We determined the main challenges inherent in recognition of Turkish handwriting as follows: delayed strokes due to a large number of diacritical marks and high OOV rates and high vocabulary similarity due to the agglutinative nature of the language. We propose removing delayed strokes and suggest and evaluate the use of sublexical stem+ending units as solutions for the OOV problem.

The experimental results show that our approach for elimination of the delayed strokes problem is effective and it increases the recognition accuracy by 2.03% points. The proposed recognition system achieves 91.7% word recognition accuracy with a middle-sized, 1,950-word lexicon with 0% OOV. As for the large vocabulary recognition task, the proposed solution of using a sub-lexical stem+ending lexicon outperforms the word-based recognition method. We achieve 67.9% recognition accuracy using a bi-gram model, with a lexicon of size 12,500 (stems and endings total). In comparison, the accuracy of the word-based lexicon approach is lower (63.1%) despite a 10-fold increase in lexicon size (130,000 words).

To improve performance, the HMM classifier is well studied both in this work and elsewhere, but new features may be added (along with more data to prevent overfitting) and this would affect the success of the following steps. Further improvements may also be possible in language modeling, by carefully choosing the stems and endings in the lexicon, to have a larger coverage on the target domain.

Acknowledgment

We gratefully acknowledge the support from the Scientific and Technological Research Council of Turkey (TÜBİTAK), under project number 113E062. The first author was also supported by a TÜBİTAK-BİDEB scholarship during her PhD studies.

References

[1] Plamondon R, Srihari SN. On-line and off-line handwriting recognition: A comprehensive survey. IEEE T Pattern Anal 2000; 22: 63-84.

[2] Al-Helali BM, Mahmoud SA. Arabic online handwriting recognition (AOHR): a survey. ACM Comput Surv 2017; 50: 33:1-33:35.

[3] Tagougui N, Kherallah M, Alimi AM. Online Arabic handwriting recognition: a survey. Int J Doc Anal Recog 2013; 16: 209-226.

[4] Doermann DS, Jaeger S. Arabic and Chinese Handwriting Recognition : Summit, SACH 2006 Selected Papers. Berlin, Germany: Springer, 2008.

[5] Plötz T, Fink GA. Markov models for offline handwriting recognition: a survey. Int J Doc Anal Recog 2009; 12: 269-298.

[6] Arıca N, Yarman-Vural FT. Optical character recognition for cursive handwriting. IEEE T Pattern Anal 2002; 24: 801-813.

[7] Hu J, Lim SG, Brown MK. Writer independent on-line handwriting recognition using an HMM approach. Pattern Recogn 2000; 33: 133-147.

[8] Biem A. Minimum classification error training for online handwriting recognition. IEEE T Pattern Anal 2006; 28: 1041-1051.

[9] Liwicki M, Bunke H. Handwriting recognition of whiteboard notes. In: Proceedings of the 12th Conference of the International Graphonomics Society; 26–29 June 2005; Salerno, Italy. pp. 118-122.

[10] Liwicki M, Bunke H. Handwriting recognition of whiteboard notes - studying the influence of training set size and type. Int J Pattern Recogn 2007; 21: 83-98.

[11] Rabiner LR. A tutorial on Hidden Markov Models and selected applications in speech recognition. P IEEE 1989; 77: 257-286.

[12] Liwicki M, Graves A, Bunke H, Schmidhuber J. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In: Proceedings of the 9th International Conference on Document Analysis and Recognition ICDAR; 23–26 September 2007; Curitiba, Brazil. pp. 367-371.

[13] Graves A, Fernández S, Liwicki M, Bunke H, Schmidhuber J. Unconstrained on-line handwriting recognition with recurrent neural networks. In: Proceedings of the 20th International Conference on Neural Information Processing Systems NIPS; 3–6 December 2007; Vancouver, BC, Canada. pp. 577-584.

[14] Graves A, Liwicki M, Fernandez S, Bertolami R, Bunke H, Schmidhuber J. A novel connectionist system for unconstrained handwriting recognition. IEEE T Pattern Anal 2009; 31: 855-868.

[15] Jäger S, Manke S, Reichert J, Waibel A. Online handwriting recognition: the NPen++ recognizer. Int J Doc Anal Recog 2001; 3: 169-180.

[16] Garcia-Salicetti S, Dorizzi B, Gallinari P, Wimmer Z. Maximum mutual information training for an online neural predictive handwritten word recognition system. Int J Doc Anal Recog 2001; 4: 56-68.

[17] Kozielski M, Doetsch P, Ney H. Improvements in RWTH's system for offline handwriting recognition. In: 12th International Conference on Document Analysis and Recognition ICDAR; 25–28 August 2013; Washington, DC, USA. pp. 935-939.

[18] Doetsch P, Hamdani M, Ney H. Gimenez A, Andres-Ferrer J, Alfons J. Comparison of Bernoulli and Gaussian HMMs using a vertical repositioning technique for offline handwriting recognition. In: International Conference on Frontiers in Handwriting Recognition ICFHR; 18–20 September 2012; Bari, Italy. pp. 3-7.

[19] Caillault É, Viard-Gaudin C. Mixed discriminant training of hybrid ANN/HMM systems for online handwritten word recognition. Int J Pattern Recogn 2007; 21: 117-134.

[20] Schenk J, Rigoll G. Novel hybrid NN/HMM modelling techniques for on-line handwriting recognition. In: Tenth International Workshop on Frontiers in Handwriting Recognition IWFHR; 23–26 October 2006; La Baule, France. pp. 619-623.

[21] Gauthier N, Artières T, Gallinari P, Dorizzi B. Strategies for combining on-line and off-line information in an on-line handwriting recognition system. In: 6th International Conference on Document Analysis and Recognition ICDAR; 10–13 September 2001; Seattle, WA, USA. pp. 412-416.

[22] Marukatat S, Artières T, Gallinari P, Dorizzi B. Sentence recognition through hybrid neuro-markovian modeling. In: 6th International Conference on Document Analysis and Recognition ICDAR; 10–13 September 2001; Seattle, WA, USA. pp. 731-737.

[23] Schenkel M, Guyon I, Henderson D. On-line cursive script recognition using time-delay neural networks and Hidden Markov Models. Mach Vision Appl 1995; 8: 215-223.

[24] Çapar A, Tasdemir K, Kılıç Ö, Gökmen M. A Turkish handprint character recognition system. In: 18th International Symposium on Computer and Information Sciences ISCIS; 3–5 November 2003; Antalya, Turkey. pp. 447-456.

[25] Kaplan K, Ertunç HM, Vardar E. Handwriting character recognision by using fuzzy logic. Fırat University Turkish Journal of Science & Technology 2017; 12: 71-77.

[26] Korkmaz SU, Kirçiçeği G, Akıncı Y, Atalay V. A character recognizer for Turkish language. In: 7th International Conference on Document Analysis and Recognition ICDAR; 3–6 August 2003; Edinburgh, Scotland, UK. pp. 1238-1241.

[27] Yanıkoğlu B, Kholmatov A. Turkish handwritten text recognition: a case of agglutinative languages. In: Document Recognition and Retrieval X DRR; 22–23 January 2003; Santa Clara, CA, USA. pp. 227-233.

[28] Şekerci M. Turkish connected and slant handwritten recognition system. MSc, Trakya University, Edirne, Turkey, 2007.

[29] Vural E, Erdoğan H, Oflazer K, Yanıkoğlu BA. An online handwriting recognition system for Turkish. In: Document Recognition and Retrieval XII DRR; 16–20 January 2005; San Jose, CA, USA. pp. 56-65.

[30] Abdelazeem S, Eraqi HM. On-line Arabic handwritten personal names recognition system based on HMM. In: 11th International Conference on Document Analysis and Recognition ICDAR; 18–21 September 2011; Beijing, China. pp. 1304-1308.

[31] Alimi AM. An evolutionary neuro-fuzzy approach to recognize on-line Arabic handwriting. In: 4th International Conference Document Analysis and Recognition ICDAR, 18–20 August 1997; Ulm, Germany. pp. 382-386.

[32] Flann NS. Recognition-based segmentation of on-line cursive handwriting. In: Proceedings of the 6th International Conference on Neural Information Processing Systems NIPS; 29 November–2 December 1993; Denver, CO, USA. pp. 777-784.

[33] Biadsy F, El-Sana J, Habash J. Online Arabic handwriting recognition using Hidden Markov Models. In: Tenth International Workshop on Frontiers in Handwriting Recognition IWFHR; 23–26 October 2006; La Baule, France.

[34] Abdelaziz I, Abdou S, Al-Barhamtoshy H. Large vocabulary Arabic online handwriting recognition system. Pattern Anal Appl 2016; 4: 1129-1141.

[35] Ghods V, Kabir E, Razzazi F. Effect of delayed strokes on the recognition of online Farsi handwriting. Pattern Recogn Lett 2013; 34: 486-491.

[36] Arısoy E, Dutağacı H, Arslan LM. A unified language model for large vocabulary continuous speech recognition of Turkish. Signal Process 2006; 86: 2844-2862.

[37] Erdoğan H, Büyük O, Oflazer K. Incorporating language constraints in sub-word based speech recognition. In: Proceedings of the 23rd Workshop of the Italian Neural Networks Society WIRN; 23–25 May 2013; Vietri sul Mare, Salerno, Italy. pp. 98-103.

[38] Sak H, Saraçlar M, Güngör T. Morphology-based and sub-word language modeling for Turkish speech recognition. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP; 14–19 March 2010; Dallas, TX, USA. pp. 5402-5405.

[39] Sak H, Saraçlar M, Güngör T. Morpholexical and discriminative language models for Turkish automatic speech recognition. IEEE T Audio Speech 2012; 20: 2341-2351.

[40] Arısoy E, Can D, Parlak S, Sak H, Saraçlar M. Turkish broadcast news transcription and retrieval. IEEE T Audio Speech 2009; 17: 874-883.

[41] Liwicki M, Bunke H, HMM-based on-line recognition of handwritten whiteboard notes. In: Tenth International Workshop on Frontiers in Handwriting Recognition IWFHR; 23–26 October 2006; La Baule, France. pp. 595-599.

[42] Günter S, Bunke H. Optimizing the number of states, training iterations and gaussians in an HMM-based handwritten word recognizer. In: 7th International Conference on Document Analysis and Recognition ICDAR; 3–6 August 2003; Edinburgh, Scotland, UK. pp. 472-476.

[43] Young SJ, Kershaw D, Odell J, Ollason D, Valtchev V, Woodland P. The HTK Book Version 3.4. Cambridge, UK: Cambridge University Press, 2006.

[44] Çöltekin Ç. A set of open source tools for Turkish natural language processing. In: Proceedings of the Ninth International Conference on Language Resources and Evaluation LREC'14; 26–31 May 2014; Reykjavik, Iceland. pp. 1079-1086.

[45] Jurafsky D, Martin JH. Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition. 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall, 2009.

[46] Weng F, Stolcke A, Sankar A. Efficient lattice representation and generation. In: Proceedings of 5th International Conference on Spoken Language Processing; 30 November–4 December 1998; Sydney, Australia. pp. 2531-2534

[47] Sak H, Güngör T, Saraçlar M. Resources for Turkish morphological processing. Lang Resour Eval 2011; 45: 249-261.

[48] Stolcke A. SRILM - an extensible language modeling toolkit. In: Proceedings of 7th International Conference on Spoken Language Processing; 16–20 September 2002; Denver, CO, USA. pp. 901-904.