Applications of Maximum Entropy Rankers to Problems in Spoken Language Processing

Richard Sproat, Keith Hall

Google, Inc

{rws,kbhall}@google.com

Abstract

We report on two applications of Maximum Entropy-based ranking models to problems of relevance to automatic speech recognition and text-to-speech synthesis. The first is stress prediction in Russian, a language with notoriously complex morphology and stress rules. The second is the classification of alphabetic non-standard words, which may be read as words (*NATO*), as letter sequences *USA*, or as a mixed (*mymsn*). For this second task we report results on English, and five other European languages.

Index Terms: MaxEnt rankers, text normalization, pronunciation modeling

10.21437/Interspeech.2014-176

1. Introduction

In this paper we report on ranking Maximum Entropy models for two applications of relevance to automatic speech recognition and text-to-speech synthesis.

The first — which we summarize here since it was previously reported in [1] — is stress prediction for Russian, which has a complex stress system that interacts with the morphology of the language. The use of SVM-based ranking models for stress prediction has been reported in prior work [2], but we argue here for the advantages of a MaxEnt approach, as well as the novel features we propose.

The second application is the classification of alphabetic non-standard words, which may be read as words (*NATO*), as letter sequences *USA*, or as a mixed (*mymsn*). To our knowledge, this work is the first to propose using a ranking model for this application. For this second task we report results on English, and five other European languages.

2. Maximum Entropy Rankers

Suppose one has a process that generates alternative analyses for a given input. Examples would be stress patterns for a word, where one might choose different syllables as being the recipient of the primary stress. Or tagging of a sequence of tokens where different tag sequences may be possible. We assume that each choice of analysis is associated with a vector of feature assignments. In each case one or more of the analyses is correct, and the rest incorrect. One can frame such a situation as a *ranking* problem where the task is to learn to rank the good analyses above the bad[1]. During training, one generates a set of alternative analyses, including the correct ones, and trains the system to discriminate among the alternatives.

In our work, we model this problem using a Maximum Entropy ranking framework similar to that presented in [3]. Suppose the problem is stress assignment (Section 3). Then, for each example, x_i, we generate the set of possible stress patterns $\mathcal{Y}_i$. Our goal is to rank the items in $\mathcal{Y}_i$ such that all of the valid stress patterns $\mathcal{Y}_i^*$ are ranked above all of the invalid stress patterns. Our objective function is the likelihood of the conditional distribution:

$$\mathcal{L} = \prod_i p(\mathcal{Y}_i^* | \mathcal{Y}_i, x_i) \quad (1)$$

$$\log \mathcal{L} = \sum_i \log p(\mathcal{Y}_i^* | \mathcal{Y}_i, x_i) \quad (2)$$

$$= \sum_i \sum_{y' \in \mathcal{Y}_i^*} \frac{\log e^{\sum_k \theta_k f_k(y', x)}}{Z} \quad (3)$$

Z is defined as the sum of the conditional likelihood over all hypothesized stress predictions for example x_i:

$$Z = \sum_{y'' \in \mathcal{Y}_i} e^{\sum_k \theta_k f_k(y'', x)} \quad (4)$$

Equation 3 defines the joint probability over an hypothesis set to be the sum of probabilities of items in that set (effectively a disjunction), which distributes mass linearly amongst items in the hypothesis set. Alternatively, we consider the expected value of probability for the hypothesis set:

$$\mathcal{L} = \sum_i \frac{\log \sum_{y' \in \mathcal{Y}_i^*} e^{\sum_k \theta_k f_k(y', x)}}{Z} \quad (5)$$

The objective function for Equation 5 is no longer convex when there is more than one item in each positive hypothesis set ($\mathcal{Y}_i^*$), Nevertheless, we use an online gradient-based optimization, stochastic gradient descent (SGD), to find a local optima for this objective. This approach follows that which is widely used for learning in Neural Networks[4]. We use a parallel SGD implementation to allow us to scale to large datasets.

During training, we provide all plausibly correct primary stress patterns as the *positive* set $\mathcal{Y}_i^*$. At prediction-time, we evaluate all possible stress predictions and pick the one with the highest score under the trained model Θ:

$$\arg\max_{y' \in \mathcal{Y}_i} p(y' | \mathcal{Y}_i) = \arg\max_{y' \in \mathcal{Y}_i} \sum_k \theta_k f_k(y', x) \quad (6)$$

The primary motivation for using Maximum Entropy rather than ranking-SVM (cf. previous work on stress prediction reported in [2]) is for efficient training and inference. Under the above Maximum Entropy model, we apply a linear model to each hypothesis (i.e., we compute the dot-product) and sort according to this score. This makes inference (prediction) fast in comparison to ranking SVM-based approaches.

[1] Alternatively, this can be seen as a *subset-selection* problem where, given a set of hypotheses, we want to select the good analyses

	acc	unacc	postacc
Dat Sg	гор'оху	г'ороду	корол'ю
	gor'oxu	g'orodu	korolj'u
Dat Pl	гор'охам	город'ам	корол'ям
	gor'oxam	gorod'am	korolj'am
	'pea'	'town'	'king'

Table 1: Examples of accented, unaccented and postaccented nouns in Russian, for dative singular and plural forms.

All experiments presented in this paper use the Iterative Parameter Mixtures parallel SGD training optimizer [5]. Under this training approach, per-iteration averaging has a regularization-like effect for sparse feature spaces.

Additionally we experiment with L1-regularization and feature-space random projections via feature hashing. We find feature-hashing to be far more effective for generating compact models for this model/optimizer combination. For example, on the Russian stress-prediction task we compact a model with 40-million active features to one with 1-million active features using feature hashing with no loss in accuracy. We use a simple form of random projection, where each parameter is assigned a id using a deterministic hash function modulo the desired size of the model[6].

The problem setting in this paper could also be modeled with a sequence model such as a Conditional Random Field (CRF). If we limited our models to non-global features for each decision, we can train a CRF where we add the global constraints (i.e., a single primary stress for stress prediction or a single switch point for letter sequence classification). Note that under these constraints, the normalization of the conditional ranking model presented here, will result in a very similar optimization. However, we find that the global features are helpful in these tasks.

3. Stress Prediction in Russian

Our work on using Maximum Entropy rankers for stress prediction in Russian has been presented in [1], so we will merely summarize the results here.

Knowing how to stress a word in Russian depends upon knowing whether the stem has an accent (and if so, where) and similarly whether the suffix has an accent [7]. If the stem is accented, any suffix accented is overridden. With unaccented stems, if the suffix has an accent, then stress for the whole word is on the suffix; if the suffix is also unstressed, then a default rule places stress on the first syllable of the word. For *postaccented* words, the accent is placed uniformly on the first syllable of the suffix. These cases can be handled by assigning an accent to the stem, indicating that it is associated with the syllable *after* the stem. Some examples of each of these classes, from [7, example 11], are given in Table 1.

Stress placement in Russian is important for speech applications since besides the phonetic effects of stress itself (prominence, duration, etc.), the position of stress strongly influences vowel quality. To take an example of the lexically unaccented noun город *gorod* 'city', the genitive singular г'орода *g'oroda* /gˈɔrədə/ contrasts with the nominative plural город'а *gorod'a* /gərʌdˈa/. All non-stressed /a/ are reduced to schwa — or by most accounts if before the stressed syllable to /ʌ/; see [8].

In previous work on stress prediction, [2] used features based on trigrams consisting of a vowel letter, the preceding consonant letter (if any) and the following consonant letter (if any). Attached to each trigram is the stress level of the trigram's vowel — 1, 2 or 0 (for no stress). For the English word *overdo* with the stress pattern 2-0-1, the basic features would be *ov:2*, *ver:0*, and *do:1*. Notating these pairs as $s_i : t_i$, where s_i is the triple, t_i is the stress pattern and i is the position in the word, the complete feature set is given in Table 2, where the stress pattern for the whole word is given in the last row as $t_1 t_2 ... t_N$. Dou and colleagues use an SVM-based ranking approach, so they generated features for all possible stress assignments for each word, assigning the highest rank to the correct assignment. The ranker was then trained to associate feature combinations to the correct ranking of alternative stress possibilities. Given the discussion above, plausible additional features are all prefixes and suffixes of the word, which might be expected to better capture some of the properties of Russian stress patterns than the much more local features from [2]. In this case for all stress variants of the word we collect prefixes of length 1 through the length of the word, and similarly for suffixes, except that for the stress symbol we treat that together with the vowel it marks as a single symbol. Thus for the word *gorod'a*, all prefixes of the word would be *g*, *go*, *gor*, *goro*, *gorod*, *gorod'a*.[2]

Substring	s_i, t_i s_i, i, t_i
Context	s_{i-1}, t_i $s_{i-1}s_i, t_i$ s_{i+1}, t_i $s_i s_{i+1}, t_i$ $s_{i-1}s_i s_{i+1}, ti$
Stress Pattern	$t_1 t_2 ... t_N$

Table 2: Features used in [2, Table 2].

vowel	а,е,и,о,у,э,ю,я,ы
stop	б,д,г,п,т,к
nasal	м,н
fricative	ф,с,ш,щ,х,з,ж
hard/soft	ъ,ь
yo	ё
semivowel	й,в
liquid	р,л
affricate	ц,ч

Table 3: Abstract phonetic classes used for constructing "abstract" versions of a word. Note that etymologically, and in some ways phonologically, в *v* behaves like a semivowel in Russian.

In addition, we include prefixes and suffixes of an "abstract" version of the word where most consonants and vowels have been replaced by a phonetic class. The mappings for these are shown in Table 3.

Our data were 2,004,044 fully inflected words with assigned stress expanded from Zaliznyak's *Grammatical Dictionary of the Russian Language* [9]. These were split randomly into 1,904,044 training examples and 100,000 test examples. The

[2]Note that in Russian the vowel ё /jɔ/ is *always* stressed, but is rarely written in text: it is usually spelled as е, whose stressed pronuncation is /(j)ɛ/. Since written е is in general ambiguous between е and ё, when we compute stress variants of a word for the purpose of ranking, we include both variants that have е and ё.

Features	1 stress	1+2 stress
shared lemmata		
Dou et al	0.972	0.965
Aff	0.987	0.985
Aff+Abstr Aff	0.987	0.985
Dou et al+Aff	0.987	0.986
Dou et al+Aff+Abstr Aff	0.987	0.986
no shared lemmata		
Dou et al	0.806	0.798
Aff	0.798	0.782
Aff+Abstr Aff	0.810	0.790
Dou et al+Aff	0.823	0.810
Dou et al+Aff+Abstr Aff	0.839	0.815

Table 4: Word accuracies for various feature combinations for both shared lemmata and no-shared lemmata conditions. The second column reports results where we consider only primary stress, the third column results where we also predict secondary stress.

100,000 test examples contain no *forms* that were found in the training data, but most of them are word forms that derive from lemmata from which some training data forms are also derived. Given the fact that Russian stress is lexically determined as discussed above, this is perfectly reasonable: in order to know how to stress a form, it is often necessary to have seen other words that share the same lemma. Nonetheless, it is also of interest to know how well the system works on words that do not share any lemmata with words in the training data. To that end, we collected a set of 248 forms that shared no lemmata with the training data. The two sets are referred to as the "shared lemmata" and "no shared lemmata" sets.

Table 4 gives word accuracy results for the different feature combinations, as follows: Dou et al's features [2]; our affix features; our affix features plus affix features based on the abstract phonetic class versions of words; Dou et al's features plus our affix features; Dou et al's features plus our affix features plus the abstract affix features.

When we consider only primary stress (column 2 in Table 4), for the "shared lemmata" test data, Dou et al's features performed the worst at 97.2% accuracy, with all feature combinations that include the affix features performing at the same level, 98.7%. For the "no shared lemmata" test data, using Dou et al's features alone achieved an accuracy of 80.6%. The affix features alone performed worse, at 79.8%, presumably because it is harder for them to generalize to unseen cases, but using the abstract affix features increased the performance to 81.0%, better than that of using Dou et al's features alone. As can be seen combining Dou et al's features with various combinations of the affix features improved the performance further.

4. Letter Sequence Classification

In [10], tokens that are not standard words or names that one might expect to find in a dictionary are termed *non-standard words* (NSWs). NSWs include digit sequences (*123* read as *one hundred twenty three*), abbreviations (*ft* for *foot*), dates (*12/12/2013*), currency amounts (*$1.30*), among many others. An important class of cases are alphabetic NSWs, some of which are read *as words* ("ASWD") such as *NATO*, some of which are read as letter sequences ("LSEQ" — *USA*) and some which are MIXED (*mymsn*), where part of the token is read as a word and part as a sequence of letters. The ASWD cases are also properly termed "acronyms" [11], though this term is often misapplied to LSEQs too. While case information is sometimes useful — e.g. in identifying *WinNT* as a MIXED and giving some indication of the location of the switch-over from word to letter reading, such examples are often also found in case-free environments, which makes the use of case unreliable. So far as we are aware, the only published work that dealt with the classification of LSEQ versus ASWD cases was reported in [10], and that uses a letter language model for the classification: see below. Also, that work provided no serious treatment of the MIXED cases.

The analysis of alphabetic tokens could be considered a simple classification task were it not for the MIXED cases, since one must predict not only *that* a case is MIXED, but also *where* in the string the switchover occurs from LSEQ to ASWD reading occurs (or the other way around).

One can represent an ASWD versus an LSEQ reading simply using a string the same length as the word consisting of either "A" or "L". Thus *CIA* is LLL whereas *DOG* is AAA. The advantage of this representation is that it allows us to represent a mixed case, and not only indicate that it is mixed, but show where the switch occurs between LSEQ and ASWD reading. Thus *GMail* would be LAAAA, and *mymsn* would be AALLL. As a practical matter we limit ourselves in this discussion to cases of the form L^+A^+ or A^+L^+. There are mixed cases like *aandb* (LAAAL), but these are rare and we do not have enough instances in our data.

Given this representation, we can then easily cast this as a ranking problem. For a token like *pticket*, the task is to determine which of the tag strings in the set {*AAAAAAA, LAAAAAA, LLAAAAA ...LLLLLLL ...AAAAALL, AAAAAAL*} is the best tag string.

The features used in the model are:

- Ngrams (length 2 to 5) of pairings of letters and their label Thus *mym:AAL* would be an trigram in *mymsn* labeled as *AALLL*. For the ngrams we pad the strings with "#" to mark the beginnings and ends of the sequences.
- Ngrams (length 2 to 5) of pairings of *abstract character classes* and the label. The classes are C(onsonant), V(owel), and Y: construction of these classes for a language requires no linguistic knowledge other than basic categorization of the letters of the alphabet. Thus for *mymsn*/AALLL, *CYC:AAL* would be one feature.
- The length of the token
- For a MIXED case, whether the ASWD portion is actually a word in the lexicon. (We use here the lexicons that are part of the Google text-to-speech system.)

Training data consisted of 314,000 labeled English tokens, with 284,384 ASWDs, 27,246 LSEQs and 2,370 MIXED cases.[3].

Results on 35,175 test examples are shown in Table 5. The second and third columns show the results with all the features (second) and excluding the abstract features (third). Using all the features provides slight gains in most cases. The only place where that is not the case is for precision for the MIXED cases, where there is a slight gain with a more restrictive set of features. However the lack of abstract features yields a substantial drop in F-measure. The fourth column contains the results of the ranking model with the exact same features as the pair language model baseline described next.

[3]Most of the ASWDs were actually words in the lexicon rather than acronyms, which is fine since most acronyms are pronounced as words in large measure because they *look* like words

	MaxEnt Ranker			Baseline	
	All	−abs	ngrm	TTS	PairLM
ASWD (P)	0.99	0.98	0.98	0.93	0.98
ASWD (R)	0.99	0.99	0.99	1.00	0.98
ASWD (F)	0.99	0.99	0.99	0.97	0.98
LSEQ (P)	0.88	0.89	0.90	1.00	0.81
LSEQ (R)	0.89	0.86	0.86	0.35	0.82
LSEQ (F)	0.89	0.88	0.88	0.68	0.81
Mixed (P)	0.76	0.78	0.68	0.00	0.42
Mixed (R)	0.60	0.45	0.50	0.00	0.39
Mixed (F)	0.68	0.61	0.59	0.00	0.41
Acc.	0.98	0.97	0.98	0.93	0.96

Table 5: Letter sequence classification for English. In the second column are shown the results using all the features, in the third column the results without the *abstract character-class* features, and in the fourth just the letter n-gram features. The righthand column shows the performance of the baselines (TTS rules, pair LM). Note that evaluation of the MIXED cases includes not only classifying them as MIXED, but also what the correct tagging for the individual letters is. If that is wrong then the example is counted as wrong.

The final two columns give two baselines. For baseline 1 we used the previous method used by the Google TTS system, namely:

- If the word is in the system's lexicon, classify it as ASWD.
- If it is one of a (small) set of known LSEQs, classify it as an LSEQ.
- If it has a vowel, classify it as ASWD.
- Otherwise classify it as LSEQ.

Note that no MIXED cases are handled. The recall for the baseline for ASWD is 1.00 since all the test ASWD cases are apparently in the TTS lexicon, which gives the baseline an obvious advantage. The precision for LSEQs is also 1.00 since the only letter sequences that get classified as LSEQ by the baseline are either in a list of known cases, or are vowelless sequences, which is guaranteed to be correct.

For baseline 2 we used a 7-gram pair language model trained over letters paired with their tags. Though the language model in general outperforms baseline 1, it underperforms the MaxEnt ranking model. In particular the MaxEnt reranking model performs far better than the pair language model on the LSEQ and MIXED cases, suggesting that the ranking model really is buying us gains over a sequence model.

Table 6 shows the application of the same approach to German, Spanish, French, Italian and Dutch. The performance on the MIXED cases is quite variable across languages and correlates somewhat ($r = 0.65$) with the amount of MIXED training examples.

5. Future Work

The stress model for Russian is in the process of being integrated into the Google Russian TTS system. We are also in the process of training stress prediction models for other languages, including Dutch and English. We are also extending the work on letter sequence classification to include Russian.

Language	de	es	fr	it	nl
# ASWD	193,433	256,688	270,010	190,582	196,678
# LSEQ	11,571	5,147	19,484	5,836	10,072
# Mixed	3,142	893	3,914	3,700	672
ASWD (P)	0.99	1.00	0.99	0.99	0.99
ASWD (R)	1.00	1.00	1.00	1.00	1.00
ASWD (F)	0.99	1.00	0.99	1.00	0.99
LSEQ (P)	0.94	0.98	0.94	0.97	0.94
LSEQ (R)	0.93	0.94	0.93	0.95	0.89
LSEQ (F)	0.94	0.96	0.93	0.96	0.92
Mixed (P)	0.82	0.78	0.72	0.81	0.75
Mixed (R)	0.67	0.50	0.52	0.62	0.21
Mixed (F)	0.75	0.64	0.62	0.72	0.48
Acc.	0.99	1.00	0.99	0.99	0.99

Table 6: Classification precision, recall, F measure for ASWD, LSEQ and Mixed cases, as well as overall accuracies for German, Spanish, French, Italian and Dutch. At the top are also given are the amounts of training data for each type for each language. Note that we used *abstract character-class* features in all cases.

6. Acknowledgements

We thank the Google Speech Data Operations team and in particular Ara Kim and Daniel van Esch for their work on preparing the data used in this research. We also thank anonymous reviewers for their comments.

7. References

[1] K. Hall and R. Sproat, "Russian stress prediction using maximum entropy ranking," in *Empirical Methods in Natural Language Processing (EMNLP)*, Seattle, WA, 2013.

[2] Q. Dou, S. Bergsma, S. Jiampojamarn, and G. Kondrak, "A ranking approach to stress prediction for letter-to-phoneme conversion," in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Suntec, Singapore: Association for Computational Linguistics, August 2009, pp. 118–126.

[3] M. Collins and T. Koo, "Discriminative reranking for natural language parsing," *Computational Linguistics*, vol. 31, pp. 25–69, March 2005.

[4] L. Bottou, "Online algorithms and stochastic approximations," in *Online Learning and Neural Networks*, D. Saad, Ed. Cambridge, UK: Cambridge University Press, 1998.

[5] K. B. Hall, S. Gilpin, and G. Mann, "Mapreduce/bigtable for distributed optimization," in *Neural Information Processing Systems Workshop on Leaning on Cores, Clusters, and Clouds*, 2010.

[6] K. Ganchev and M. Dredze, "Small statistical models by random feature mixing," in *Proceedings of the ACL-08: HLT Workshop on Mobile Language Processing*. Columbus, Ohio: Association for Computational Linguistics, June 2008, pp. 19–20.

[7] M. Halle, "On stress and accent in Indo-European," *Language*, vol. 73, no. 2, pp. 275–313, 1997.

[8] T. Wade, *A Comprehensive Russian Grammar*. Oxford: Blackwell, 1992.

[9] A. Zaliznyak, *Grammaticheskij slovar' russkogo jazyka*. Moscow: Russkiy Yazik, 1977.

[10] R. Sproat, A. Black, S. Chen, S. Kumar, M. Ostendorf, and C. Richards, "Normalization of non-standard words." *Computer Speech and Language*, vol. 15, no. 3, pp. 287–333, 2001.

[11] G. Cannon, "Abbreviations and acronyms in English word-formation," *American Speech*, vol. 64, pp. 99–127, 1989.