Learning Phrase Patterns for ASR Name Error Detection Using Semantic Similarity

Alex Marin[1,2], *Mari Ostendorf*[2], *Ji He*[2]

[1]Microsoft Corporation, Bellevue, WA U.S.A.
[2]University of Washington, Seattle, WA U.S.A.

alex.marin@microsoft.com, ostendor@uw.edu, jvking@uw.edu

Abstract

This paper addresses the problem of detecting name errors in automatic speech recognition (ASR) output, when available ASR output training data is sparse and covers only a limited subset of scenarios from a broad-domain task. We reduce the dimensionality of the lexical feature set using a related task (sentence-level name detection) and improve feature coverage by expanding single words to classes using semantic similarity measures derived from both WordNet and neural network word embeddings. Phrase patterns are learned over the selected word features using a frequency-based pattern selection algorithm. In experiments on English dialogs, we find that adding sentence-level features performs better than using local n-gram context. Automatically-learned seed features perform better than hand-crafted patterns, and their class expansions generalize better. Finally, the embedding-based class expansions yield better results than the corresponding WordNet-based configurations.

Index Terms: speech recognition, name error detection, word embeddings, word class features

10.21437/Interspeech.2015-341

1. Introduction

In many spoken language processing systems, automatic speech recognition (ASR) errors pose significant problems, both to human users reading the resulting transcriptions and for automatic systems (e.g. machine translation, information extraction). Misrecognized names form an important category of errors, often due to the fact that names are more likely to be out-of-vocabulary (OOV). Names also tend to be important sources of information, so more costly to miss. If name errors can be detected, the system may take steps to correct the error regions, either automatically, for example by transliteration from the recognized phone sequence, or through user input.

Early work on name error detection incorporated ASR word confidence estimates in a named entity recognition (NER) system [1, 2, 3], taking advantage of local contextual cues to names (e.g. titles for person names). Parada *et al.* [4] extend this work by applying the NER tagger to a word confusion network based on a hybrid word/fragment ASR system. Hatmi *et al.* [5] integrate NER directly into the ASR decoder via a generic NE token in the language model. Kumar *et al.* [6] experiment with different methods of integrating ASR and NER with a maximum entropy classifier and part-of-speech features. He *et al.* take a similar approach with somewhat different features, including both local and sentence-level lexical features as well as confusion network structural cues to errors. Chen *et al.* [7] propose a variable-span approach to directly tagging longer spans of consecutive words instead of individual words.

The problem of name error detection specifically (vs. named entity detection) has the added challenge of highly skewed priors: name errors are infrequent. One approach to dealing with this problem is to hold out names from the ASR vocabulary to artificially increase the number of errors for feature learning, and later tune detection thresholds for more realistic priors. However, in domains with limited training resources, it is difficult to learn sufficient lexical cues to names even with this approach. This paper investigates methods to address the lexical feature learning problem by deriving word classes that generalize a set of seed features learned from limited training data. Building on the work in [8], which uses a sentence-level named entity predictor to learn features related to the presence or absence of a name in an utterance, word classes associated with those seed features are derived using various semantic similarity measures to specify a neighborhood of words. The use of word classes allows us to keep the feature dimensionality low, while enriching the set of lexical sequences which may indicate the presence of a name, thus ameliorating the data sparsity problem. We experiment with both external, domain-agnostic sources of semantic similarity, in the form of WordNet and broad-domain data-driven embeddings, and with the use of text from the target domain to learn embeddings.

2. Task and System Overview

The name error detection system used in this paper is a component of a bidirectional speech-to-speech translation system for English-Iraqi Arabic discussions, with human-computer dialogs to resolve detected errors and ambiguities. The data consists of a range of conversations; the primary topics include military and humanitarian assistance and training, though many other topics are included [9]. The task often involves mentions of people and places. The focus of this paper is on detecting regions of the ASR output which correspond to errors overlapping a name in the human transcripts of the data, whether the name is OOV or not. Experiments are based on the English side source data.

2.1. Name Error Detection

We perform binary classification over ASR word confusion networks (WCN) [10], with each WCN slot marked as positive if it overlaps with a name in the reference string and negative otherwise. We use a maximum entropy (MaxEnt) classifier with L2 regularization, built using the MALLET toolkit [11]. The baseline system uses three types of features: confusion network structural features extracted over the full WCN for an utterance, and word confidences and word n-gram context features, extracted over the 1-best path of the WCN for that utterance.

Confusion network structural features describe the distribution of arcs in their corresponding WCN slots, using the intu-

ition that slots with more arcs correspond to confusable regions in the input signal. The features include the highest posterior for a given slot, the number of arcs in a slot, and the mean and standard deviation of the slot arc posteriors. The full set of features is described in [12]. The word confidence features (from [13]) provide a posterior indicating the likelihood that the highest-scoring word in each WCN slot is correct. The baseline lexical context features consist of unigrams, bigrams, and trigrams over the 1-best path through a WCN. They are intended to provide cues that distinguish names from other ASR error categories.

While we perform classification at the WCN slot level, we report results with contiguous sequences of error slots aggregated into a single error region. In this paper, we report region-level F-score results, which capture whether the detected name error regions overlap true name errors.

2.2. Datasets

The datasets used in this paper were developed during the DARPA TRANSTAC and BOLT projects. A set of 8781 spoken utterances are available for training the name error detection systems. These are split 60-20-20 into internal training (TRAIN), development (DEV), and evaluation (INT EVAL) sets. As in [7, 8], we decode the internal training and development sets with a reduced vocabulary to increase the rate of name errors. A set of in-domain language model training data (LM-Train) collected during the TRANSTAC [14] project, consisting of 1.6 million sentences (15.4 million words) is used for learning word embeddings. The DEV set is used for feature selection and decision threshold tuning. The INT EVAL set is used for selecting the best word class expansion configurations (i.e. max class size) and exploring the impact of domain variation. The final evaluation is performed on a test set designed by NIST (referred to as NIST EVAL), consisting of 1890 utterances, with at least 20% of the utterances on topics not seen in the training set.

3. Semantic Similarity Feature Learning

We design a new method to learn task-oriented lexical features based on word-level semantic similarity measures. Unlike purely supervised algorithms, which incorporate the task objective directly into the feature learning, we achieve the supervision in a first step of seeding the algorithm with discriminative words or phrase patterns learned or hand-specified using the internal training set. The features defined by the seed set are generalized to word classes to improve coverage using different semantic similarity measures, with part-of-speech (POS) filtering applied to preserve the syntactic structure of the seeds. The feature space dimensionality remains unchanged. In this work, we use automatic POS tagging using the parser described in [15], with output tags matching the Penn Treebank [16] tagset.

3.1. Initial Seed Feature Definition

3.1.1. Hand-crafted Seed Features

The first approach uses a human-generated list of features likely to indicate the presence or absence of names in an utterance, based on inspection of a subset of the utterances in the TRAIN set. We use a set of about 80 features, with each feature matching one of a small number of syntactic templates. A few representative examples of the seed features, grouped according to the sequence of corresponding POS tags, are shown in table 1. We also include in the table the count of features belonging to each group. Only a few hand-crafted seed features contain more than two words (e.g. *the people of*). We explicitly allow gaps in all multiword features, thus treating them as phrase patterns.

POS Group	Count	Examples
VB* PP	34	start in, looking for, studying in
NN	8	captain, lieutenant, tribe
PRP* NN	7	my name, your rank, your neighbor
DET NN	5	the school, the street, the area

Table 1: Examples of hand crafted seed features grouped by their corresponding POS sequences.

3.1.2. Automatically-Generated Seed Features

As automatically-generated seed features we use the features learned by an auxiliary MaxEnt classifier trained on reference transcripts of the TRAIN dataset, targeting the presence or absence of *names* (not name errors) in the sentence, as described in [8]. For this auxiliary classifier, each utterance is labeled as positive if it contains at least one name in the reference transcript. As classification features the system uses unigram features extracted over the sentence, excluding names. A feature selection step based on information gain (i.e. mutual information with the class variable), tuned on the DEV set, led to this classifier using the top 170 words.

An additional, optional step involves the generation of phrase patterns from the list of unigram features, using an algorithm similar to the phrase pattern learning approach described in [17]. We map the seed words onto the utterances in the TRAIN dataset, collecting sequences of such seed words (not necessarily contiguous) and their frequency. Infrequent patterns are pruned out, with the frequency threshold set to select the top 80 patterns, matching the number of patterns used in the hand-crafted seed feature set. Some representative seed patterns used in this configuration include: *is ... name*, *how ... in*, and *have ... in*. The overlap with the hand-crafted patterns is minimal, since most of those words either are not present in the automatically-derived 170 word set, or appear only in infrequent automatically-generated patterns.

3.2. Semantic Feature Expansion

We expand the set of features using a neighborhood-based approach. We first define the word class for a single seed word as the semantic neighborhood of the word, i.e. the set of words closest to the seed in terms of a pre-defined semantic relatedness function *sim*; the expansion set of a multi-word pattern is defined as the cartesian product of the class expansions of each individual word. The size of the semantic similarity class for a word is restricted to select only matches of high similarity.

Many researchers have used WordNet [18] as a source of semantic relatedness for various NLP tasks [19, 20, 21], and different similarity metrics have been defined to take advantage of the WordNet ontology. In this work, we use Lin's similarity measure [22], which provides a score with range $[0, 1]$.

An alternative method to obtain semantic similarity information is through word embeddings [23]. We experiment with two types of word embeddings, learned using the RNNLM toolkit described in [24] and the *word2vec* skip-gram algorithm [25], respectively. To obtain a similarity score from the word embeddings, we compute the Euclidean distance $d(x, y)$ between the two embedding vectors x and y. The distance is used to obtain a similarity function with a range of $[0, 1]$ defined as $sim(x, y) = 2(1 - d'(x, y))$, where $d'(x, y) = 1/(1 + \exp(-d(x, y)))$.

The analysis of the hand-crafted seed patterns shows that a large proportion of the features can be grouped into a small number of syntactic templates. To preserve the syntactic structure of the seeds in the extended feature set, we apply POS filtering to the neighborhoods obtained using semantic similarity measures, with the words selected as possible class members being required to match the POS tag of the seed word. The matching uses a relaxed POS definition, with related POS tags from the Penn Treebank [16] set grouped into a broader POS category as shown in table 2; POS tags not shown are preserved as individual tags. While any given word may have multiple POS tags depending on context, the expansion classes are context-independent, chosen to be the most frequent POS tag according to the distribution in the LMTRAIN corpus. The tags are generated automatically using a POS tagger developed by Nasr et al. [15] for a related project. Preliminary experiments show that POS-based filtering provides some benefit in terms of reducing spurious matches and improves performance [12].

POS Group	POS Members
noun	NN, NNS
proper noun	NNP, NNPS
pronoun	PRP, PRP$
adjective	JJ, JJR, JJS
adverb	RB, RBR, RBS
verb	MD, VB, VBD, VBG, VBN, VBP, VBZ
w-pronoun	WDT, WP, WP$, WRB
preposition	IN, TO

Table 2: POS tag groups used in POS filtering.

3.3. Feature Extraction

Even after word class expansion, the feature set size remains fixed, determined by the initial seed set. In the class representation of the feature space, we define each feature based on the word classes corresponding to members of the initial seed set. Each feature takes positive value if it is observed in the 1-best path of the confusion network ASR output. During feature extraction, if any combination of words from the classes corresponding to a seed feature are present in the data in the order specified by the seed feature, the corresponding class pattern-based feature is activated; otherwise the feature has value 0.

4. Experiments

We experiment with features based on three different seed sets, each of which replaces the baseline lexical context feature set in the classifier setup.

- **Hand-crafted** - the set of 80 hand-annotated phrase pattern features.
- **Auto words** - all 170 words with non-zero weight in the sentence-level name detector.
- **Auto patterns** - the set of the 80 most frequent patterns extracted automatically over the set of **auto words**.

4.1. The Role of Long-Distance Context

In the first set of experiments, we examine the impact of using features contiguous to the slot (i.e. *local*) vs. features extracted over the entire sentence (i.e. *sentence*-level). We also evaluate the impact of different types of lexical features. For both local and sentence-level features we experiment with single word as well as n-gram features, both restricted to the **Auto words** seed set. Additionally, for local features we use the full set of n-gram context features, not restricted to the **Auto words** word list. We use the set of **Auto patterns** seed features as a comparison point when we extract features over the whole sentence. Finally, we add a configuration that combined the full set of n-gram context features and the **Auto words** features.

Context	Feature Set	Feature Count	INT Eval	NIST Eval
Sent.	Auto words	170	47.2	45.0
	Auto n-grams	850	44.8	**45.2**
	Auto patterns	80	34.1	**45.2**
Local	Auto words	170	49.1	41.2
	Auto n-grams	850	38.2	40.0
	all n-grams	146k	53.7	40.0
Sent.+local	Auto words + all n-grams	$\sim$146k	**49.6**	35.3

Table 3: F-score results for automatically-learned seed features using different forms of context.

Results are presented in table 3. The last two rows are roughly comparable to the best systems reported in [8], with minor regularization differences. The best case results on the INT EVAL set use all local n-gram features, but these cases (using vastly more features) correspond to the worst results on the NIST EVAL set. We hypothesize that this is due to the fact that the TRAIN and DEV sets more closely match the INT EVAL set; the NIST EVAL data includes scenarios/topics that are not covered in the training data. The sentence-level lexical features seem to generalize better to the NIST EVAL data than the local versions of these features. The **Auto patterns** have mixed results, perhaps because of the small number of features.

4.2. Feature Expansion Experiments

Next, we compare the effect of the various feature expansion methods proposed, when used to augment each of the three seed feature sets described above. For each seed set configuration, we include results using only the seed words or patterns, as well as configurations which use the expanded word classes to enrich the feature space. The semantic feature expansion configurations include the WordNet-based similarity and similarities based on four different word embeddings (table 4).

Similarity	Algorithm	Training Set	Init. Val.
RNN-BOLT	RNNLM	LMTrain	random
SG-BOLT	skip-gram	LMTrain	random
SG-GN	skip-gram	GN	random
SG-Both	skip-gram	GN+LMTrain	[25]+random

Table 4: Word embedding-based similarity configurations.

The **RNN-BOLT** and **SG-BOLT** configurations are both trained using only in-domain LM training data, using RNNLM and skip-gram embeddings, respectively. The **SG-GN** configuration uses Google News (GN) data for training a skip-gram embedding, with words not found in the GN vocabulary estimated using the average of the embeddings for words in their local context. The **SG-Both** configuration uses the in-domain LM data to train a skip-gram embedding, initializing with the GN embeddings in [25] and random values for words not covered by the GN vocabulary. The maximum class size of the word embedding-based classes is tuned over the range $[5, 80]$, with similarity thresholds so most classes have under 50 members before class size thresholding. For classes learned us-

ing WordNet similarity, we found that many classes consisted of members y with semantic similarity to the seed word x, $\text{sim}(x, y) = 1$. Thus, we experimented with only two configurations: using only matches with maximal similarity, or setting the maximal class size to $\max(50, |\{y : \text{sim}(x, y) = 1\}|)$. The WordNet similarity does not expand expand classes associated with function words, unlike the embeddings-based similarities.

Seed Set	Similarity	Ave. |Class|	INT EVAL	NIST EVAL
Hand-crafted	Seed Only	1	33.5	36.5
	WordNet	33	41.8	42.3
	RNN-BOLT	19	44.1	40.6
	SG-BOLT	5	43.7	**42.4**
	SG-GN	13	**44.3**	38.9
	SG-Both	13	42.7	37.5
Auto words	Seed Only	1	47.2	45.0
	WordNet	22	46.0	45.9
	RNN-BOLT	19	51.5	40.2
	SG-BOLT	5	49.9	**47.0**
	SG-GN	10	49.5	36.6
	SG-Both	6	**53.0**	39.6
Auto patterns	Seed Only	1	34.1	45.2
	WordNet	33	39.0	42.2
	RNN-BOLT	15	**43.7**	**47.1**
	SG-BOLT	4	40.8	39.2
	SG-GN	4	40.0	41.5
	SG-Both	11	39.5	44.3

Table 5: F-score results for different seed set configurations and both semantic similarity sources.

F-score results are shown in table 5. For each seed feature set, semantic feature expansion yields benefits in almost all cases for the INT EVAL set. Performance on the NIST EVAL set is mixed, though at least one semantic class expansion configuration outperforms the corresponding seed-only configuration in each case. In general, some in-domain data is useful, with best results being obtained using the **SG-BOLT** configuration for the **Hand-crafted** and **Auto words** seed sets, and the **RNN-BOLT** configuration for the **Auto patterns** seed set. The WordNet-based classes yield competitive results, though they are never the best. The best configurations on the INT EVAL set do not always predict the best results on the NIST EVAL set, though WordNet tends to be the most consistent across the two sets. Examining class sizes, we find that WordNet classes are largest over all similarity sources, due to the large number of matches with maximal similarity; we also notice larger classes in the expansions of the **Hand-crafted** seed set. Results using a modified WER metric [12] designed to measure the extent of error regions showed only negligible differences between the various configurations and are not included here.

Figure 1 presents the DET curves for three systems over the false alarm (FA) region [0.02, 0.2], using only local n-gram context (green), auto-word seeds (red), and auto-words with **SG-BOLT** expansion (blue). In the operating region of interest (FA=[0.05,0.1]), the class expansion slightly outperforms the other two configurations, though over the full range the configurations using auto word features perform similarly. Both outperform the local n-gram context configuration.

We inspected several classes obtained from different similarity measures to gain insights into the various approaches. To illustrate, table 6 contains the top-ranked matches for the seed word *street* found using each semantic similarity algorithm. We

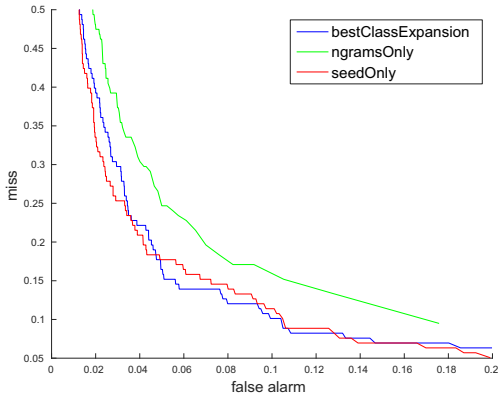

Figure 1: Miss vs. false alarm rates for auto word seed and best class expansion features vs. baseline n-grams (NIST EVAL).

find that, perhaps not surprisingly, the matches obtained from WordNet are like the seed word in terms of their ontological type; for example, *street* is like other kinds of objects which people use as land transportation surfaces. On the other hand, **RNNLM-BOLT** matches are similar to the seed in terms of some intrinsic property; for example, for the seed word *street*, that property seems to be that all the matches are other types of locations. The two skip-gram embeddings that use the Google n-grams data are similar to WordNet; on the other hand, the **SG-BOLT** similarity, trained using only in-domain data, yields noisier results, with the top matches being more similar to the RNNLM than to the generic domain classes, but also containing some outliers, such as *fourteenth* or *currant*. Since there seem to be performance advantages to using in-domain data, further exploration of methods to combine the different sources might lead to better name error detection performance.

Similarity	Most Similar Matches to "street"
WordNet	thoroughfare, routes, roads, way, highway
RNN-BOLT	city, pharmacy, corner, restaurant, field
SG-BOLT	suburb, fourteenth, currant, road, embassies
SG-GN	streets, avenue, terrace, road
SG-Both	road, sidewalk, alley, highway, corner

Table 6: Sample semantic similarity feature classes.

5. Discussion

In this paper, we investigated methods for improving the automatic detection of ASR name errors, focusing on learning features that are robust to sparse training conditions. We explored different levels of context, finding that long-distance features improve performance over using only local n-gram context cues. Our feature expansion based on word classes learned using word-level semantic similarity provides a small additional gain, in particular when the similarity is obtained from in-domain-trained word embeddings. Future work will explore additional word class learning strategies, aiming to more effectively combine broad domain and target domain data. Another possible direction is to leverage syntactic and discourse cues.

Acknowledgments

This material is based on work supported by DARPA under Contract No. HR0011-12-C-0016 (subcontract 27-001389). Any opinions, findings and conclusions or recommendations expressed herein are those of the authors and do not necessarily reflect the views of DARPA.

6. References

[1] D. Palmer, M. Ostendorf, and J. Burger, "Robust information extraction from automatically generated speech transcriptions," *Speech Communication*, vol. 32, pp. 95–109, 2000.

[2] D. Palmer and M. Ostendorf, "Improving out-of-vocabulary name resolution," *Computer Speech and language*, vol. 19, no. 1, pp. 107–128, 2005.

[3] K. Sudoh, H. Tsukada, and H. Isozaki, "Incorporating speech recognition confidence into discriminative named entity recognition of speech," in *Proc. ACL*, 2006.

[4] C. Parada, M. Dredze, and F. Jelinek, "OOV sensitive named-entity recognition in speech," in *Proc. Interspeech*, 2011, pp. 2085–2088.

[5] M. Hatmi, C. Jacquin, E. Morin, S. Meigner *et al.*, "Incorporating named entity recognition into the speech transcription process," in *Proc. Interspeech*, 2013.

[6] R. Kumar, R. Prasad, S. Ananthakrishnan, A. N. Vembu, D. Stallard, S. Tsakalidis, and P. Natarajan, "Detecting OOV named-entities in conversational speech," in *Proc. Interspeech*, 2012.

[7] W. Chen, S. Ananthakrishnan, R. Prasad, and P. Natarajan, "Variable-span out-of-vocabulary named entity detection." in *Proc. Interspeech*, 2013, pp. 3761–3765.

[8] J. He, A. Marin, and M. Ostendorf, "Effective data-driven feature learning for detecting name errors in automatic speech recognition," in *SLT*, 2014.

[9] N. Ayan, A. Mandal, M. Frandsen, J. Zheng, P. Blasco, A. Kathol, F. Bechet, B. Favre, A. Marin, T. Kwiatkowski, M. Ostendorf, L. Zettlemoyer, P. Salletmayr, J. Hirschberg, and S. Stoyanchev, ""Can you give me another word for hyperbaric?" Improving speech translation using targeted clarification questions," in *Proc. ICASSP*, 2013, pp. 8391–8395.

[10] L. Mangu, E. Brill, and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," *Computer Speech & Language*, vol. 14, no. 4, pp. 373–400, 2000.

[11] A. K. McCallum, "MALLET: A machine learning for language toolkit," 2002, http://mallet.cs.umass.edu.

[12] M. A. Marin, "Effective use of cross-domain parsing in automatic speech recognition and error detection," Ph.D. dissertation, University of Washington, 2015.

[13] Y.-C. Tam, Y. Lei, J. Zheng, and W. Wang, "ASR error detection using recurrent neural network language model and complementary ASR," in *Proc. ICASSP*, 2014, pp. 2312–2316.

[14] Translation technology: Breaking the language barrier. Http://www.darpa.mil/workarea/downloadasset.aspx?id=2676.

[15] A. Nasr, F. Béchet, J. Rey, B. Favre, and J. Le Roux, "MACAON: an NLP tool suite for processing word lattices," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Systems Demonstrations*. Association for Computational Linguistics, 2011, pp. 86–91.

[16] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger, "The Penn treebank: Annotating predicate argument structure," in *In ARPA Human Language Technology Workshop*, 1994, pp. 114–119.

[17] A. Marin, R. Holenstein, R. Sarikaya, and M. Ostendorf, "Learning phrase patterns for text classification using a knowledge graph and unlabeled data," in *Proc. Interspeech*, 2014.

[18] G. A. Miller and et al., "WordNet: An on-line lexical database," *Intl. Journal of Lexicography*, vol. 3, pp. 235–244, 1990. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.105.1244

[19] Z. Wu and M. Palmer, "Verbs semantics and lexical selection," in *Proc. ACL*, 1994.

[20] M. Mohler and R. Mihalcea, "Text-to-text semantic similarity for automatic short answer grading," in *Proc. EACL*, 1994.

[21] A. Marin, W. Wu, B. Zhang, and M. Ostendorf, "Detecting targets of alignment moves in multiparty discussions," in *Proc. ICASSP*, 2012.

[22] D. Lin, "An information-theoretic definition of similarity," in *Proc. International Conference on Machine Learning*, 1998.

[23] T. Mikolov, W.-T. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proc. NAACL-HLT*. Association for Computational Linguistics, 2013.

[24] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Cernockỳ, "RNNLM-recurrent neural network language modeling toolkit," in *Proc. ASRU*, 2011, pp. 196–201.

[25] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013. [Online]. Available: http://arxiv.org/abs/1301.3781