How to Add Word Classes to the Kaldi Speech Recognition Toolkit

Axel Horndasch, Caroline Kaufhold, and Elmar Nöth

Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU),
Lehrstuhl für Informatik 5 (Mustererkennung),
Martensstraße 3, 91058 Erlangen, Germany
`{axel.horndasch,caroline.kaufhold,elmar.noeth}@fau.de`
`http://www5.cs.fau.de/`

Abstract. The paper explains and illustrates how the concept of word classes can be added to the widely used open-source speech recognition toolkit Kaldi. The suggested extensions to existing Kaldi recipes are limited to the word-level grammar (G) and the pronunciation lexicon (L) models. The implementation to modify the weighted finite state transducers employed in Kaldi makes use of the OpenFST library. In experiments on small and mid-sized corpora with vocabulary sizes of 1.5K and 5.5K respectively a slight improvement of the word error rate is observed when the approach is tested with (hand-crafted) word classes. Furthermore it is shown that the introduction of sub-word unit models for open word classes can help to robustly detect and classify out-of-vocabulary words without impairing word recognition accuracy.

Key words: Word classes, Kaldi speech recognition toolkit, OOV detection and classification

1 Introduction

It is a well-known fact that class-based n-gram language models help to cope with the problem of sparse data in language modeling and can reduce test set perplexity as well as the word error rate (WER) in automatic speech recognition [1] [2] [3]. But word classes can also be used to support the detection of out-of-vocabulary (OOV) words. For example in [4] multiple Part-of-Speech (POS) and automatically derived word classes are introduced to better model the contextual relationship between OOVs and the neighboring words. The authors of [5] and [6] focus on semantically motivated word classes which are combined with generic or more generalized word models. In both cases the idea is to focus on open word classes like person or location names for which OOVs are very common.

In so called hierarchical OOV models a sub-word unit (SWU) language model for OOV detection is embedded into a word-based language model; the SWU-based language model can also be inserted into a word class. This approach has been suggested for example in [7] and just recently in [8]. With the solution presented in this paper it is possible to implement a similar strategy.

The speech recognition toolkit we based our work on is the widely used open-source software suite Kaldi [9]. It provides libraries and run-time programs for state-of-the algorithms under an open license as well as complete recipes for building speech recognition systems. All training and decoding algorithms in Kaldi make use of Weighted Finite State Transducers (WFSTs), the fundamentals of which are described in [10]. By modifying the standard Kaldi transducers using the OpenFST library tools [11] we were able to integrate word classes into the decoder, a feature that was missing in Kaldi so far.

Because the topic[1] has been discussed by Kaldi users now and again, we thought it would be worthwhile to write a paper in which we share the experiences we made when we extended the Kaldi recipes to include word classes.

The rest of the paper is structured in the following way: In section 2 we introduce word classes for automatic speech recognition in a bit more detail. How word classes can be modeled in Kaldi is described in section 3. Our experiments and the data sets we used are presented in section 4 and the paper ends with a conclusion and an outlook.

2 Word Classes for Automatic Speech Recognition

2.1 Mathematical Formalism

In the context of this research we only look at non-overlapping word classes which can be defined as a mapping $C : \mathcal{W} \to \mathcal{C}$ which determines a sequence of word classes $\mathbf{c}$ given a sequence of words $\mathbf{w}$ (definition taken from [12]):

$$\mathbf{w} = w_1 \dots w_n \rightsquigarrow C(w_1) \dots C(w_n) = c_1 \dots c_n = \mathbf{c} \tag{1}$$

When using word classes the probabilities of the language model for word sequences $\mathbf{w}$ have to be adjusted. In the case of bigram modeling the formula

$$P(\mathbf{w}) = P(w_1) \prod_{i=2}^{m} P(w_i | w_{i-1}) \tag{2}$$

needs to be rewritten as

$$P(\mathbf{w}) = P(w_1 | c_1) P(c_1) \prod_{i=2}^{m} P(w_i | c_i) P(c_i | c_{i-1}) \tag{3}$$

Following the maximum likelihood principle, the class-related probability of a word $P(w_i|c_i)$ can simply be estimated by counting the number of occurrences of the word divided by the number of all words in the word class (uniform modeling). The (conditional) class probabilities $P(c_i| \dots)$ can be determined in the same way normal n-gram probabilities are computed. The only difference

[1] See for example `https://sourceforge.net/p/kaldi/discussion/1355348/thread/c7c5e4f6/`

is that the word sequences used for training must be converted to word class sequences.

The class mappings used for the data sets in this paper (see section 4) were created manually, but there are many clustering algorithms to automatically find optimal word (equivalence) classes based on measures like (least) mutual information etc. (see again [1], [2] and [3]).

2.2 Open vs. Closed Word Classes

Specially designed word classes can be used to limit the number of possible user utterances that need to be considered in the speech recognition module of a spoken dialog system. If it can be assumed that users are cooperative, this will improve recognition accuracy. Imagine for example the task of having to recognize time information like "10:40 am" in an utterance. One way to configure the language model (in essence a grammar) could be, to introduce (closed) word classes which model hours and minutes:

...	10	40	am	...
...	`NUMBER_1_TO_12`	`NUMBER_0_TO_59`	`AM_OR_PM`	...

In case some (valid) numbers were not seen in the data which was collected to train the speech recognizer, they can be easily added to the language model by putting them into the appropriate word class as an entry. Adding all possible entries for a closed word class will prevent errors caused by the out-of-vocabulary problem. There are of course many more examples for (maybe more intuitive) closed words classes e.g. `WEEKDAY`, `MONTH` etc.

For word classes with a virtually unlimited vocabulary it is impossible to rule out the OOV problem. Such open word classes come into play if a task makes it necessary to recognize named entities like person or location names for example. Nevertheless, word classes are helpful in this case too because specialized word models (generic HMMs, sub-word unit language models) can be embedded in them to capture unknown words. This approach is also called hierarchical OOV modeling and has been the subject of quite a number of publications (e.g. [5], [6], [7] and [8]). In the following section, we show how this can be implemented as part of a Kaldi recipe.

3 Modeling Word Classes in Kaldi

The Kaldi Speech Recognition Toolkit [9] uses Weighted Finite State Transducers (WFSTs) to bridge the gap between word chains and feature vectors. Four different levels of transducers are used to do that: a word-level grammar or language model G to model the probabilities of word chains, a pronunciation lexicon L which provides the transition from letter to phone sequences, a context-dependency transducer C which maps context-independent to context-dependent phones and an HMM transducer H to map context-dependent phones

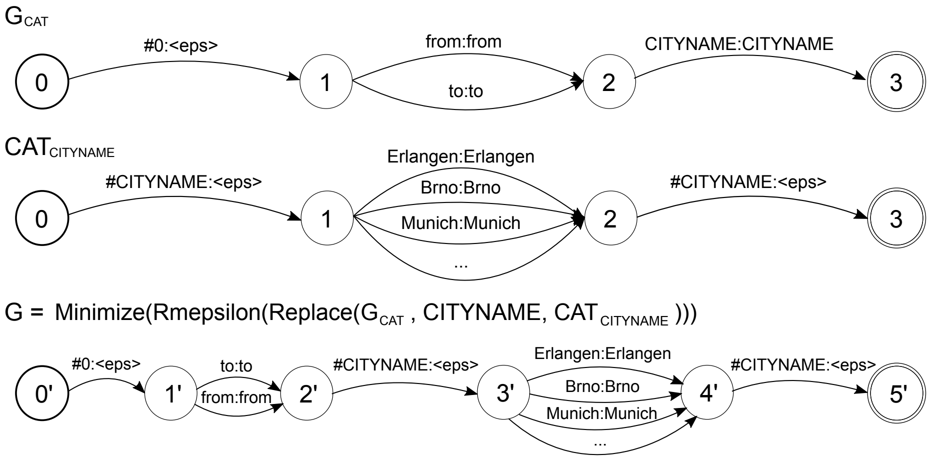

Fig. 1. The root grammar G_{CAT}, the sub-language model $CAT_{CITYNAME}$ and the resulting graph G after the replacement, `<eps>` removal and minimization steps

to so called *transition IDs* (please refer to [9] for more details). Our approach to introduce word classes in Kaldi only affects the G and L transducers in the decoding graphs.

3.1 Modifying the Kaldi Transducers

The procedure we came up with to integrate a word class into the existing Kaldi transducers is as follows:

1. Create a language model transducer G_{CAT} with class entries mapped to non-terminal string identifiers (in Figure 1 that identifier is `CITYNAME`)
2. Create sub-language models for each word class including the following steps
   – Add transitions with a class-specific disambiguation symbol as input and the empty word (`<eps>`) as output symbol before and after the actual sub-language model (in Figure 1 the disambiguation symbol is `#CITYNAME`)
   – Make sure there is no path through the sub-language model which generates no output (i.e. just empty words)
   – Convert the sub-language model to a WFST
3. Insert the sub-language model WFSTs for the according non-terminal string identifiers in G_{CAT} using `fstreplace`
4. Remove all transitions with `<eps>:<eps>` labels (`fstrmepsilon`) and minimize the resulting graph (`fstminimize`) to get G
5. Add self-loops for all class-specific disambiguation symbols to the lexicon transducer L so the word classes “survive” the composition of G and L
6. Compose G and L and carry on in the usual way

It is important to note that the non-terminal string identifier (`CITYNAME` in Figure 1) and the class-specific disambiguation symbol (`#CITYNAME` in Figure 1), which is introduced to keep the decoding graph determinizable, must not be confused. As the name indicates the non-terminal symbol is not present in the final decoding graph any more. The new disambiguation symbol however is essential during the decoding process.

3.2 Adding a Sub-Word Unit Based OOV Model

The addition of a sub-word unit (SWU) based OOV model to a word class as visualized in Figure 2 is very similar to adding a word class to the parent language model as described in section 3.1. However, there is one small but important difference: since we ideally want to remain in an SWU loop as long as an out-of-vocabulary word is encountered in the input, there is a transition back to the entry loop of the OOV model. For this transition another disambiguation symbol is needed (in Figure 2 the back transition is labeled `#CITYNAME_SWU_BACK:<eps>`).

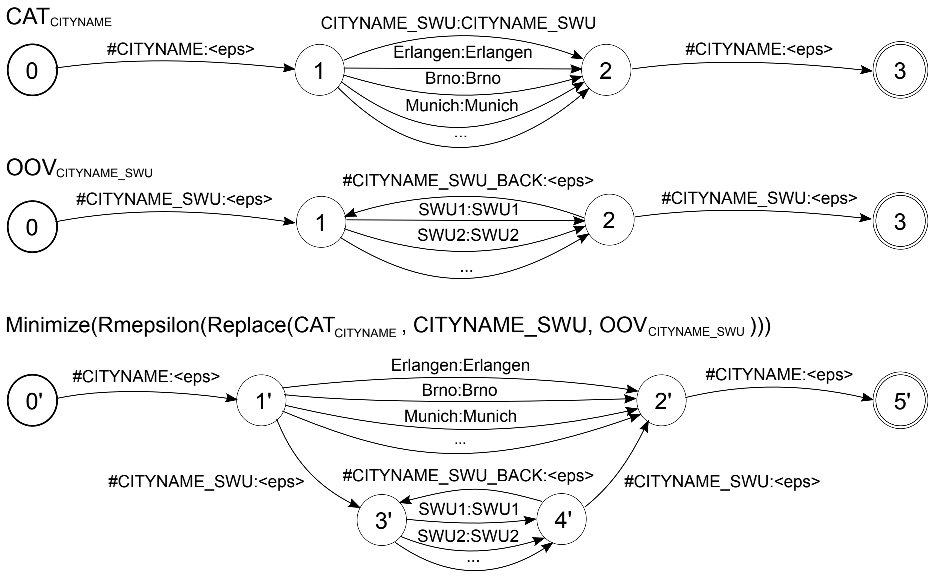

Fig. 2. The sub-language model $CAT_{CITYNAME}$ (which also serves as the root in this case), the OOV model $OOV_{CITYNAME\_SWU}$ and the resulting graph after the replacement, `<eps>` removal and minimization steps

At this point it should be noted that, after inserting an SWU-based OOV model in the proposed way, the stochasticity of the resulting WFST can not be guaranteed any more. Even though this should still be investigated further, it seems as if it doesn't have a bad effect on recognition results.

4 Data Sets and Experiments

4.1 EVAR train information system

The first data set we used for our experiments in this paper was collected during the development of the automatic spoken dialog telephone system EVAR [13]. Just as in [14] we use the subset of 12,500 utterances (10,000 for training and 2,500 for testing) which are the recordings of users talking to the live system over a phone line. The EVAR corpus used in [6] contains more utterances, but the additional data consists of read data which was initially used to train the speech recognition module and which is quite different from the spontaneous user inquiries.

Table 1 shows that the vocabulary of the EVAR corpus is limited to 1,603 words or 1,221 syllables. It also indicates the low number of words per utterance on average (less than 3.5). Nevertheless the data is suitable for our experiments because it contains the open word class CITYNAME.

Set	**Utterances**	**Words**		**Syllables**	
		Types	Tokens	Types	Tokens
Training	10,000	1,423	34,934	1,118	55,874
Test	2,500	712	8,286	669	13,154
All	12,500	1,603	43,220	1,221	69,028

Table 1. Statistics regarding words and syllables in the training and test set of the EVAR corpus

The importance of the word class CITYNAME for the EVAR corpus can be seen in Table 2: more than an eighth of all words in the test set are city names; 78 city names in the test set are out-of-vocabulary, which is almost one third of all OOV tokens. Overall the OOV rate of 2.98% is not particularly high, but this is not surprising given the functionality of the system, which was limited to providing the schedule of express trains within Germany. As expected though, the OOV rate for the open word class CITYNAME[2] is much higher (7.05%). The OOV rate for syllables (1.19%) is much lower than for words. Syllables are also the type of sub-word unit which was chosen for the OOV model for the experiments in this paper. Single phones, for which there is no OOV problem any more, have the drawback that the recognition accuracy goes down and if used in a hierarchical OOV model, they often produce very short inaccurate OOV hypotheses.

[2] To save resources it was decided during the design phase of EVAR that the system should only be able to provide information about express trains (so called IC/ICE trains). As a consequence only city names with an express train station were included in the vocabulary of the recognizer. While this may seem intuitive at first, it lead to a large number of OOVs because even cooperative users were not always sure in which cities there were express train stops.

SWU / Word Class	**Types**		**Tokens**		
	#OOVs	#all	#OOVs	#all	%
Words	180	712	247	8,286	2.98
Syllables	103	669	156	13,160	1.19
Phones	0	43	0	37,610	0.00
`CITYNAME`	41	115	78	1,106	7.05

Table 2. OOV statistics regarding words, syllables, phones as well as entries of the category `CITYNAME` in the test set of the EVAR corpus.

To test our approach we used three different speech recognizers and compared the resulting word error rates (WER); for our hierarchical OOV model we also looked at recall (percentage of OOVs found), precision (percentage of correct OOV hypotheses) and FAR (false alarm rate, the number of false OOV hypotheses divided by the number of in-vocabulary words):

1. *Baseline*: the standard s5 recipe of the Kaldi toolkit [9] with a tri-gram language model
2. *Category-based*: like baseline but with the word class `CITYNAME` added to the decoding graph following the procedure described in section 3.1
3. *Category-based + OOV*: like category-based but with a syllable-based OOV model inserted into the word class `CITYNAME`

The results of our tests on the EVAR corpus are summarized in Table 3. It can be seen that both category-based approaches – for EVAR the only word class we used was `CITYNAME` – improve the word error rate compared to the baseline recognizer. If an OOV model is included, the WER goes down even further. The reason for this is that, if no OOV model is present, the speech recognizer often tries to cover the out-of-vocabulary word with short in-vocabulary words. For example the German town "Heilbronn", which was not in the training data and thus out-of-vocabulary, was recognized as two words "Halle Bonn" by the conventional word recognizer. The recognizer which featured an OOV model for the word class `CITYNAME` produced the result "halp#Un_ORTSNAME.OOV" consisting of the two phonetic syllables "halp" and "Un".

Model	**WER**	**OOV Results**			**OOV Results (`CITYNAME`)**		
		Recall	Precision	FAR	Recall	Precision	FAR
Baseline	14.7	0.0	0.0	0.00	0.0	0.0	0.0
Category-based	14.5	0.0	0.0	0.00	0.0	0.0	0.0
Category-based + OOV	14.1	21.0	75.0	0.02	58.0	63.0	3.0

Table 3. Word error rates and OOV detection results (recall, precision, false alarm rate) for all OOVs and for out-of-vocabulary city names on the test set of the EVAR corpus.

4.2 SmartWeb Handheld Corpus

Model	**WER**	**OOV Results**			**OOV Results** (CELEBRITY)		
		Recall	Precision	FAR	Recall	Precision	FAR
Baseline	19.7	0.0	0.0	0.00	0.0	0.0	0.0
Category-based	18.8	0.0	0.0	0.00	0.0	0.0	0.0
Category-based + OOV	18.4	23.0	83.0	0.00	36.0	70.0	7.0

Table 4. Word error rates and OOV detection results (recall, precision, false alarm rate) for all OOVs and for out-of-vocabulary names of celebrities on the test set of the SmartWeb corpus.

The second data set, which we used to test our approach, is the *SmartWeb Handheld Corpus*. It was collected during the SmartWeb project, which was carried out from 2004 to 2007 by a consortium of academic and industrial partners [15]. The recordings of the handheld corpus were carried out using cell phones and the signals underwent *a complex chain of speech transmissions including Bluetooth and UMTS* [16]. The resulting data was given one of three labels: *normal*, *bad* or *unusable*. For the experiments in this paper only the *normal* recordings were used.

The SmartWeb handheld corpus is more interesting because SmartWeb was designed to be an open-domain question-answering system. As a consequence there are many more open word classes than in Evar and the vocabulary is much larger as well (5,787 word types); most out-of-vocabulary words are encountered for the class CELEBRITY (59 word tokens). Overall the OOV rate for SmartWeb on the word level is 4,86% (432 out of 8887 test tokens), on the syllable level however it's only 0,60% (104 out of 17,225 test tokens).

In Table 4 the reduction of the word error rate for both category-based recognizers can be observed again. Also, as for Evar, the very robust OOV detection can be found in the resulting data. The False Alarm Rate (FAR) of 0.00 is not a mistake: only 20 OOV hypotheses were inaccurate, on the other hand there are 8,455 in-vocabulary words. For the experiments on the SmartWeb corpus 14 categories were used for the two enhanced recognizers; the OOV model was again based on the syllables which were extracted from the training set.

5 Conclusion and Outlook

In this paper we have shown how the concept of word classes can be integrated into the Kaldi speech recognition toolkit by modifying the Weighted Finite State Transducers (WFSTs) for the language model G and the pronunciation lexicon L in Kaldi's standard recipes. In experiments on the Evar and SmartWeb data sets we showed that this approach can help improve the word error rate. Another advantage of the proposed method is the option to add an OOV model based on sub-word units to robustly detect out-of-vocabulary words for open word classes.

While we are certain that our approach will also work for vocabulary sizes beyond 5.5K words, this still needs to be proven by further experiments. Another point that should be investigated in more depth, even though it didn't impair the recognition performance, is the non-stochasticity of the resulting decoding graph after inserting the OOV model into the sub-language model. Furthermore, to improve the system, it could be explored which other sub-word units apart from syllables are suitable for the hierarchical OOV model. An obvious candidate are data-driven phonetic sub-word units which can be generated automatically based on existing pronunciation lexicons.

References

1. Ney, Hermann and Essen, Ute: On Smoothing Techniques for Bigram-Based Natural Language Modelling. In: Proceedings of International Conference on Acoustics, Speech and Signal Processing, ICASSP 1991, Toronto, Canada (1991)
2. Brown, Peter F. et al.: Class-based N-gram Models of Natural Language. In: Computational Linguistics, volume 18, pages 467–479, MIT Press, Cambridge, MA, USA(1992)
3. Kneser, Reinhard and Ney, Hermann: Improved clustering techniques for class-based statistical language modelling. In: Proceedings of EUROSPEECH 1993, Berlin, Germany (1993)
4. Bazzi, Issam and Glass James R.: A multi-class approach for modelling out-of-vocabulary words. In: Proceedings of INTERSPEECH 2002, Denver, Colorado, USA (2002)
5. Schaaf, Thomas: Detection of OOV words using generalized word models and a semantic class language model. In: Proceedings of INTERSPEECH 2001, pages 2581–2584, Aalborg, Denmark (2001)
6. Gallwitz, Florian: Integrated Stochastic Models for Spontaneous Speech Recognition. PhD Thesis, Pattern Recognition Lab, Computer Science Department 5, University of Erlangen-Nuremberg, Logos Verlag, Berlin, Germany (2002)
7. Seneff, Stephanie and Wang, Chao and Hetherington, I. Lee and Chung, Grace: A dynamic vocabulary spoken dialogue interface. In: Proceedings of INTERSPEECH 2004 Jeju Island, Korea, (2004)
8. Aleksic, Petar S. and Allauzen, Cyril and Elson, David and Kracun, Aleksandar and Casado, Diego Melendo and Moreno, Pedro J.: Improved recognition of contact names in voice commands In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015, pages 5172–5175, South Brisbane, Queensland, Australia (2015)
9. Povey, Daniel et al.: The Kaldi Speech Recognition Toolkit In: IEEE 2011 Workshop on Automatic Speech Recognition and Understanding. IEEE Signal Processing Society (2011)
10. Mohri, Mehryar and Pereira, Fernando and Riley, Michael: Weighted finite-state transducers in speech recognition. In: Computer Speech & Language, volume 16, issue 1, pages 69–88, Elsevier (2002)
11. Allauzen, Cyril et al.: OpenFst: A general and efficient weighted finite-state transducer library. In: Implementation and Application of Automata, pp 11-23, Springer-Verlag, Berlin, Germany (2007)

12. Schukat-Talamazzini, E.G.: Automatische Spracherkennung – Grundlagen, statistische Modelle und effiziente Algorithmen. Vieweg, Braunschweig, Germany (1995)
13. Eckert, W. and Kuhn, T. and Niemann, H. and Rieck, S. and Scheuer, A. and Schukat-Talamazzini, E.G.: A Spoken Dialogue System for German Intercity Train Timetable Inquiries. In: Proceedings of EUROSPEECH 1993, pages 1871–1874, Berlin, Germany (1993)
14. Stemmer, G.: Modeling Variability in Speech Recognition. PhD Thesis, Pattern Recognition Lab, Computer Science Department 5, University of Erlangen-Nuremberg, Logos Verlag, Berlin, Germany (2005)
15. Wahlster, W.: SmartWeb: Mobile Applications of the Semantic Web. GI Jahrestagung (1), pages 26–27, (2004)
16. Mögele, Hannes and Kaiser, Moritz and Schiel, Florian: SmartWeb UMTS Speech Data Collection: The SmartWeb Handheld Corpus. Procedings of LREC 2006, pages 2106–2111 Genova, Italy (2006)