DYNAMIC VOCABULARY WITH A KALDI SPEECH RECOGNIZER IN A SPEECH DIALOG SYSTEM FOR AUTOMOTIVE INFOTAINMENT APPLICATIONS

Thomas Ranzenberger, Christian Hacker, Karl Weilhammer

Elektrobit Automotive GmbH
thomas.ranzenberger@elektrobit.com

Abstract: In this paper we present an evaluation of the Kaldi speech recognizer using dynamic vocabulary in an automotive context. We updated our previously integrated Kaldi speech recognizer and make use of a new available decoding method together with a new special type of weighted finite state transducer that allows us to evaluate the usage of dynamic vocabulary. We use an existing Kaldi reference model for English and extend it to recognize names of a contact list and create a second model to recognize radio stations with a language model reduced to words for this specific domain. The contact list models are based on the librispeech corpus with two hundred thousand words and will be extended with forty, eighty and one hundred twenty words. We measured the time for modifying the reference model with the dynamic vocabulary. It took fourteen seconds for the biggest vocabulary model with additional one hundred twenty words. We tested the word error rate of the models on the librispeech corpus. The word error rates did not significantly change in comparison to our reference model. We extended the processing of the recognition result to detect the slots and match them with a list of slots. We evaluated the sentence error rate, slot detection error rate and intent error rate of the forty words contact list and the radio station model. 10 participants spoke 25 random sentences of a self created corpus of example sentences. All participants were non-native speakers. The sentences contained words of the librispeech corpus. Common names and stations of the united states were added which were not in the baseline librispeech language model. For the radio station model we used an out of vocabulary placeholder in our sentences to test the intent mapping. The contact list model with forty words had a sentence error rate of 52.00%, a slot detection error rate of 34.80% and a intent detection error rate of 11.60%. The participants had problems with the pronunciation of the country and region specific names which might origin also from outside of the united states. The domain specific radio station model had a sentence error rate of 20.40%, a slot detection error rate of 8.00% and a intent detection error rate of 1.60%. Most stations were spoken letter by letter. The high slot recognition rate and intent recognition rate of the model is caused by a reduced vocabulary for the specific domain.

1 Introduction

The use of dynamic vocabulary in the automotive domain has already been established since many years. In this paper we continue to evaluate the open source speech recognizer *Kaldi* [1] in an automotive context. We improve our previously described integration of the Kaldi speech recognizer into an existing dialog system [2]. In the following sections we show how to extend our previously trained time delayed neuronal network model with dynamic vocabulary. We

integrate a method to detect the slots within the recognition result. We describe our experiments for a contact list and a radio station list automotive scenario. For each scenario we extend the previously trained TDNN model. We verify the word error rate of the extended models on the librispeech corpus [3] and test them on specially prepared speech recordings. Finally, we focus on the evaluation of the scenarios and describe the sentence and slot error rates for each scenario.

2 Extension of Kaldi models with dynamic vocabulary

Kaldi uses weighted finite-state transducers (WFSTs [4]) to combine the acoustic model with the language model. Four WFSTs are used and assembled together. The grammar WFST **G** contains the language model which models the probabilities of word chains. A corresponding lexicon of the words which combines the orthographies of the words with the corresponding phone sequences is stored in the **L** WFST. The context is modeled using a **C** WFST. It maps the context-independent phones to the context-dependent phones. The **H** WFST is used to map the context-dependent phones to transition IDs [1]. The framework to modify the vocabulary of an trained model is described in the Kaldi documentation [5]. The basic idea is to add a nonterminal symbol like #nonterm:contact_list in the grammar WFST which is acting like a variable in the decoding graph to use the additional vocabulary. With version *4c39ce2136645dd1de9b42512b2dfe82a2d06fa7* and newer of the Kaldi repository the framework supports a new WFST GrammarFST, which can be used for this purpose. The current revision of the code in the repository is version *1dcdf80c587190de9ae189a8c2bfd071e98ca9c9* [6]. A GrammarFST is decoded using 64-bit state IDs which are interpreted as two 32-bit integers: The state IDs for a base WFST instance and for an additional WFST instance. The additional WFST instance contains the extended vocabulary and the base instance contains the state IDs of the previously trained model. During decoding, the GrammarFST will test that the final-probability for a specific state has a special value. If the value is present in the current state it will expand (compute the vector of arcs leaving it) the state with the additional states organized by the second part of the 64-bit state ID [5].

3 Approach

We use the following procedure based on the grammar examples in the recipe folder for a smaller part of the librispeech corpus of Kaldi for the modification of our existing time delayed neuronal network Kaldi model. The grammar example scripts are located in the folder *egs/mini_librispeech/local/grammar* [5]. A similar approach to add word classes for Kaldi is described in [7].

The list of nonterminals (nonterminals.txt) of the previously created dictionary is extended with a new nonterminal symbol (#nonterm:unk). We prepare a new language baseline using the modified dictionary. As a next step we use the original arpa language model of the Kaldi model and replace the <UNK> symbol with the new nonterminal. The modified arpa model is used to create a new grammar WFST (*G.fst*). The output symbols of the new nonterminal are removed. After that we create the graph for the baseline WFST using the language baseline with the newly created grammar WFST. For the newly added vocabulary we create a pronunciation lexicon by hand or by using a dictionary like the CMU Pronouncing Dictionary [8] to lookup the pronunciation for the new vocabulary. We keep the default probability for the pronunciations. We create a new grammar WFST for the additional words. Each word will be assigned with equal probability. The grammar WFST Table 1 uses nonterminal symbols which are enabled for output. Each word will be produced going from state 1 to state 2. The last line in the table sets

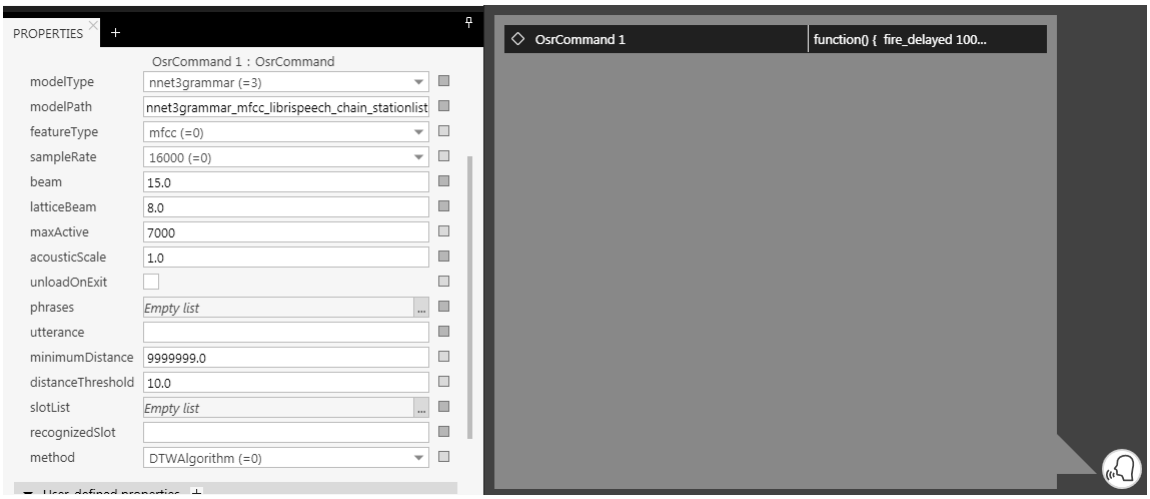

Figure 1 – Talk with one OsrCommand-Spidget to recognize slots. On the left side you see properties of the selected spidget, i.e. *phrases* with a list of example phrases and *utterance* with a link to the global datapool, where the recognized utterance is written into during runtime. The usage of multiple OsrCommand-Spidgets enables the mapping of intents[2]. The *slotList* contains a list of slots to be recognized and is linked to the global datapool. The *recognizedSlot* property stores the text of the recognized slot and is linked to the global datapool to be able to use the value of the slot in other talks.

the terminal state of the WFST to state 3. After the creation of the grammar WFST we compile

Table 1 – Grammar WFST which produces the extended vocabulary

StartState	EndState	InputSymbol	Output
0	1	#nonterm_begin	#nonterm_begin
2	3	#nonterm_end	#nonterm_end
3			

a decoding graph WFST (*HCLG.fst*) for the extended vocabulary part which will be included in the previously created baseline graph (*HCLG.fst*). Both graphs are combined by using Kaldi's *make-grammar-fst* command. We write the resulting graph as a GrammarFST by enabling the parameter –write-as-grammar [5]. A already fully productive set up with replacement of word classes for intents during runtime is described by [9].

3.1 Extended OSR recognizer and OsrCommand-Spidget

In our previous paper we created a Kaldi based recognizer OSR. We extended the decoding part of the recognizer to be able to decode GrammarFST's and added a new parameter to the socket connection for the parameter *mode*. The version *b50a4cf4dfcf1a8abbb5c643ac0cce75d1dfa6b3* of the master branch of the Kaldi framework was used to update the decoding code of our implementation [10]. By sending the parameter with the value *nnet3grammar* we enable the decoding of GrammarFST files for the models used in our scenarios. The output of the recognizer contains the nonterminals of the WFST in Table 1. This enables the result processing algorithm to detect the slot for the new vocabulary. The result processing is done by a spidget [11] which is available in our human-machine-interaction modeling tool EB GUIDE Studio [11]. We extended the spidget to specify the new GrammarFST recognition mode. We further added a *slotList* property and a property which contains the recognized slot values (see Figure 1). The result processing in the OsrCommand-Spidget uses the n-best-list with the nonterminals, extracts the corresponding words (slot values) and compares them with the content of the *slotList*. If a corresponding value is found in the slot list, the result will be propagated in the recognized-

Slot property of the spidget. In the following section we describe the scenarios which should be used for our experiments to test the usage of dynamic vocabulary within our speech dialog system.

3.2 Automotive scenario

We focus on two scenarios for our approach. In the first scenario the user wants to call another person or send a message to someone. In this scenario a contact list with names is used in the car to find the corresponding phone numbers or e-mail addresses. The recognition system needs to update the list of contacts if a new smartphone is paired or a cloud account is activated. In the second scenario the user wants to listen to the radio. The radio station list is changing while the position of the car is changing. If the car leaves a specific region, new stations appear and old stations disappear. In this scenario the speech recognition system needs to recognize the new list of currently available station names.

4 Experiment

For the contact list scenario we use census data of the united states to create lists of contacts with twenty (model name Contactlist20), forty (model name Contactlist40) and sixty (model name Contactlist60) entries. For the contact list scenario we extend the baseline language model `tgsmall` of 200.000 words of the librispeech corpus [3] with the first-name male [12], female [13] and last-names [14] of the census data which were not included in the language model before. The `tgsmall` language model is based on tri-grams (`tg` and has still an acceptable word error rate. We use the procedure which is described in section 2 to create models which extend the baseline language model with forty, eighty and one hundred twenty words. The forty contact list entries model contains ten male first-names [12], ten female first-names [13] and twenty last-names [14]. For the evaluation we choose the following random names: Alisa McDowell, Alisa McGee, Brianna McMillan, Cordell Meza, Daren McKay, Darren McMillan, Dylan McKenzie, Dylan Zavala, Elvis McDowell, Jacquelyn Esquivel, Jacquelyn McClain, Jacquelyn O'Brien, Jeanine Avalos, Jeanine O'Brien, Jerri McDaniel, Kasey Zavala, Leonel Magana, Leonel O'Brien, Marcelino Esquivel, Sheri McMahon.

For the radio station list scenario we slightly modify the procedure in section 2. We estimated a separate language model which was trained on in-domain sentences for radio station commands. This highly specialized language model, which contained only 27 words was used as the baseline language model for the second scenario. The new baseline language model is extended by radio stations in the united states. The stations are added by choosing the three top radio stations for each state of the united states [15]. We use twenty five station names to extend the baseline language model and create the StationList1 Kaldi model. The list of radio stations contains the following entries: WCBS, WBEN, WABC, KSUR, KNX, KARY, KNHC, KNDD, WPBB, WANM, WFLZ, KITY, KRLD, KNCT, KAFF, KSED, KSSK, KORL, KQQL, KFXN, KMNB, KKLZ, KUNV, KDWN, WBZ. We created the pronunciations for the station names manually by adding letter by letter pronunciations from the CMU Pronouncing Dictionary [8] and also used the embedded sub-words of the stations to add pronunciation alternatives. It is also possible to to address sequences of duplicate letters by using the word *double* e.g. *double d*. We created the described Kaldi models on a virtual machine. In a first experiment we measured the time for the modification of the Kaldi models. The results of the experiment are shown in Table 2.

As a second experiment we evaluated the word error rate (WER) of the contact list models on the librispeech corpus and compared it to the unchanged reference TDNN model. Table

Table 2 – Table with modification time of the models with word counts for the automotive scenarios

Modelname	Count of baseline words	Count of expanded words	Modification time
Contactlist20	200.000 words LM	40 words	12 seconds
Contactlist40	200.000 words LM	80 words	12 seconds
Contactlist60	200.000 words LM	120 words	14 seconds
StationList1	27 words LM	25 words	7 seconds

Table 3 shows a decline of 0.34 % for the contact list models in both test sets of the librispeech corpus.

Table 3 – Word error rates (WER) of the reference TDNN model and the contact list scenario models for the librispeech corpus. Data indicates the used audio data (clean vs. other) of the librispeech corpus and the language model (tg: tri-gram, small/large: size after pruning).

Data	WER			
	TDNN	ContactList20	ContactList40	ContactList60
test_clean_tgsmall	5.91%	5.93%	5.93%	5.93%
test_other_tgsmall	14.80%	14.84%	14.84%	14.84%

We use an own automotive corpus *EB-Car* for our next experiments. We create random sentences based on the corpus and the mentioned slot values for the contact list and radio station list scenarios. We use the StationList1 and the ContactList20 model for our evaluation. Sentences contain intents with slots like *call <CONTACT_NAME>* or *play <STATION_NAME>*. The words for the intent sentences of *EB-Car* are a subset of the words which are used for the librispeech corpus based Kaldi model. The words which are in the slots were added to the baseline models as described before. We used one part of the *EB-Car* corpus sentences only for the training of the statistical language model. The other part of the *EB-Car* sentences is used for the evaluation. The test sentences for the radio station scenario contain one out of vocabulary word. This adds additional difficulty to the mapping of the intent and will lower the sentence recognition rate for this specific experiment. Each participant of our evaluation was asked to speak 25 random sentences out of the *EB-Car* corpus for each model. The participants are using a headset. We use a sample rate of 16 kHz. We set for each scenario the *beam* parameter for the decoding to 15 and the *lattice beam* to 8. We use a maximum of 7000 active states for the decoding. The acoustic scale parameter is set to one. The configuration of the parameters for one evaluation model in the modeling environment is shown in Figure 1. The DTW algorithm compares the recognized utterance without the slots with the scenario specific set of intent phrases from the *EB-Car* corpus. We count the sentences correctly identified by the DTW algorithm. Finally, we compare percentage of correctly recognized sentences from the OSR with the percentage of correctly recognized intents after DTW and mapping to one of the intent phrases of the corpus.

We extended our evaluation model described in [2] to calculate the statistics for the number of slots and intents (DTW). It counts each intent or slot which is mapped to the correct entries of the slot list or intent (phrases) list of the used OsrCommand-Spidget. Figure 2 shows the extended evaluation model during runtime. The participant starts a recognition of the given sentence by pressing the push to talk (*PTT*) button. If the recognition is finished the participant is able to display the next sentence by pressing the *Next* button. At the same time the statistics are updated.

Figure 2 – Evaluation model during runtime. Buttons for the workflow are on the left side. The csv export options are on the right side. On the top the sentence is displayed and the first recognition result of the recognizer (nBestList) as well as the result after the processing of the DTW algorithm (utterance) and the result for the matching slot. The statistic shows the accumulated number of evaluated sentences (# sentences) and accumulated numbers for the recognizer, slots and the DTW algorithm for intent mapping (# OK and # FAILED).

5 Evaluation

For the evaluation we had 9 male and 1 female participants. None of the participants was a participant of the librispeech corpus [3] collection. All participants were non-native speakers. The evaluation was conducted in English. For each model of the experiment we recorded two hundred fifty sentences. The Table 4 shows the error rates in percent. The sentence error rate

Table 4 – Evaluation of the sentence error rate of the recognizer (SER) slot detection error rate (SDER) and intent detection error rate (IDER) of the Kaldi models for the automotive scenario

model	SER	SDER	IDER
ContactList20	52.00%	34.80%	11.60%
StationList1	20.40%	8.00%	1.60%

for the ContactList20 is much higher as the rate for the StationList1 model. The ContactList20 model has a large vocabulary and may recognize other words which sound similar to the contact list names. The contact names are rare set of names which are from different cultures or even states outside the united states. The pronunciation of the names for non-native speakers is very hard. Another reason for the low recognition rate is that the word *EMAIL* and the intent *FORWARD TO* seem to have low probability in the baseline statistical language model. 65.20% of the slots were detected correctly. In our previous paper the reference TDNN model had a sentence error rate of 52.52% [2]. The value of the ContactList20 model did just slightly change. The intent recognition rate is 88.40%. The DTW algorithm used for the intent mapping works still good, even if the recognition rate of 48.0% is very low.

The StationList1 sentence error rate of 20.40% is a result of the out of vocabulary (OOV) word which was added to the test sentences. The word *digital* in the sentence *switch on digital radio station KNCT* was recognized for example as *switch on <SPOKEN_NOISE> radio*

station #nonterm_begin KNCT #nonterm_end. The OOV word was recognized as unknown sequence, in this case modeled as <*SPOKEN_NOISE*> in the language model. The slot and intent recognition is still working in this example. 199 sentences out of 250 sentences were recognized correctly. The slot detection rate is 92.00%. The slot error rate of 8.00% is a result of a similar pronunciation of station names like *WABC* and *WABZ* of the non-native speakers. The slot matching algorithm uses the first n-best-list result and matches it with the slot list. If the slot list contains the slot it will propagate it to the speech dialog system. If the second best recognition result contains the correct slot it will not match anymore. The intent recognition rate for the radio station intent phrases is 98.40%.

6 Conclusion

In this paper we evaluated the usage of dynamic vocabulary in the Kaldi toolkit for automotive scenarios. We showed how to extend Kaldi models with dynamic vocabulary and extended our decoder to decode these models to provide recognized utterances and slot values to the speech dialog system. We used the dynamic time warping (DTW) algorithm to detect intents and added an additional slot result processing method which provides the recognized slot to the speech dialog system. We showed that the extended dynamic vocabulary models do not significantly lower the word error rate in comparison to the reference model based on TDNN with a word error rate of 5.91% on the test data with a tri-gram language model.

We used an EB GUIDE model [11] to evaluate the extended models for a contact list and a radio station list scenario. 10 people participated on our evaluation to speak 25 random sentences for each scenario. The sentence error rate for the contact list scenario with an vocabulary of 200.040 words was 52.00%. The error rate for the radio station list scenario with 52 words vocabulary (27 baseline language model vocabulary extended by 25 radio station names) was 20.40%. We measured the slot detection error rate (SDER). The contact list model had 34.80% SDER and the radio station list model 8.00%. We measured the ability of the DTW algorithm to map to intents for both models. The intent detection error rate of the contact list model was 11.60%. The rate for the radio station list model was 1.6%. The non-native speakers had a hard task to speak the specific names of the contacts and pronounce the radio station names correctly. The reduction of the vocabulary to a specific domain improves the sentence error rate and results in better slot and intent detection.

References

[1] POVEY, D., A. GHOSHAL, G. BOULIANNE, L. BURGET, O. GLEMBEK, N. GOEL, M. HANNEMANN, P. MOTLICEK, Y. QIAN, P. SCHWARZ, J. SILOVSKY, G. STEMMER, and K. VESELY: *The kaldi speech recognition toolkit*. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, 2011. URL `http://publications.idiap.ch/downloads/papers/2012/Povey_ASRU2011_2011.pdf`.

[2] RANZENBERGER, T., C. HACKER, F. GALLWITZ, and N. GERMANY: *Integration of a kaldi speech recognizer into a speech dialog system for automotive infotainment applications*. In *Conference on Electronic Speech Signal Processing (ESSV 2018), Ulm*. 2018.

[3] PANAYOTOV, V., G. CHEN, D. POVEY, and S. KHUDANPUR: *Librispeech: An asr corpus based on public domain audio books*. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210. 2015. doi:10.1109/ICASSP.2015.7178964.

[4] MOHRI, M., F. PEREIRA, and M. RILEY: *Weighted finite-state transducers in speech recognition*. 2002.

[5] *Support for grammars and graphs with on-the-fly parts.* 2019. URL http://kaldi-asr.org/doc/grammar.html. Accessed on 24 January 2019 08:32 pm.

[6] *History for src/decoder/grammar-fst.h kaldi-asr/kaldi github*. 2019. URL https://github.com/kaldi-asr/kaldi/commits/master/src/decoder/grammar-fst.h. Accessed on 27 January 2019 08:20 pm.

[7] HORNDASCH, A., C. KAUFHOLD, and E. NÖTH: *How to add word classes to the kaldi speech recognition toolkit*. *Text, Speech, and Dialogue*, 2016. doi:10.1007/978-3-319-45510-5_56. URL http://dx.doi.org/10.1007/978-3-319-45510-5_56.

[8] LENZO, K.: *The cmu pronouncing dictionary*. *Carnegie Melon University*, 2007.

[9] COUCKE, A., A. SAADE, A. BALL, T. BLUCHE, A. CAULIER, D. LEROY, C. DOUMOURO, T. GISSELBRECHT, F. CALTAGIRONE, T. LAVRIL ET AL.: *Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces*. *arXiv preprint arXiv:1805.10190*, 2018.

[10] *Github kaldi-asr/kaldi*. 2019. URL https://github.com/kaldi-asr/kaldi. Accessed on 27 January 2019 08:46 pm.

[11] MASSONIE, D., C. HACKER, and T. SOWA: *Modeling graphical and speech user interfaces with widgets and spidgets*. *ITG-Fachbericht: Speech Communication*, (252), 2014. URL https://www.vde-verlag.de/proceedings-de/453640016.html. VDE Verlag GmbH, Berlin/Offenbach, ISBN 978-3-8007-3640-9.

[12] *Most common male names in the u.s.* 2019. URL https://names.mongabay.com/male_names_alpha.htm. Accessed on 25 January 2019 04:06 pm.

[13] *Top female first names in america*. 2019. URL https://names.mongabay.com/femal_names_alpha.htm. Accessed on 25 January 2019 04:05 pm.

[14] *What is the most common last name in the united states*. 2019. URL https://names.mongabay.com/data/1000.html. Accessed on 25 January 2019 04:04 pm.

[15] *Search for u.s. radio stations by state*. 2019. URL https://radio-locator.com/cgi-bin/page?p=states. Accessed on 25 January 2019 04:06 pm.