SPOKEN LANGUAGE UNDERSTANDING IN EMBEDDED SYSTEMS

Karl Weilhammer, Prince Kumar, Volker Springer and Dominique Massonie

Elektrobit Automotive, Am Wolfsmantel 46, 91058 Erlangen, Germany
{Karl.Weilhammer,Prince.Kumar,Volker.Springer,Dominique.Massonie}@elektrobit.com

Abstract: We investigated classification with Support Vector Machines for spoken language understanding with respect to their use in embedded devices, which are often equipped with slow CPUs, and main or persistent memory of limited size or with slow access times. We started with uni- and bigrams as features and managed to reduce the feature set in most cases by applying Recursive Feature Elimination from a few thousands to a few dozens. This corresponds to a reduction of the overall model size to 6% of the original size, without having a substantial loss in classification performance. The F score difference is 0.16%.

1 Introduction

In recent years, speech dialog systems (SDS) have become a natural component in embedded devices such as mobile phones, smart watches and cars. *Elektrobit Automotive* integrates SDS for major car manufacturers such as Audi, General Motors and others. In an increasing number of systems parts of the processing are done on a server, e.g. automatic speech recognition (ASR). Unfortunately, in many situations the mobile network is not available. Imagine, a driver uses underground parking and wants to issue a speech command to conveniently set up the navigation system before starting the trip or think about driving in a remote area with unreliable network connection. In addition to these technical limitations an increasing number of users are concerned about their privacy and feel uneasy when their speech commands are sent to a server and potentially collected there. All this shows that there is a demand for speech and dialog processing on device. Embedded software devices are usually characterized by slow CPUs with a small cache. They are often equipped with main or persistent memory of limited size or with slow access times. We describe how techniques for spoken language understanding (SLU) can be optimized with respect to small model size and fast processing time, while making minimal concessions to accuracy and reliability of the classifier. A good overview of recent developments in the field of SLU is given in [1]. We will briefly describe those developments that are relevant to our work.

1.1 Grammar based models

The classic SLU approach is grammar based word spotting, where the rules are written by experts [2][3]. This works well for small systems and ad hoc solutions. Rule based models are small and processing times are fast. But for complex systems which are deployed in many languages and have a huge number of rules, hand crafted systems come in less handy as for each language an expert linguist is necessary to handle the complicated task of writing and maintaining the grammar.

1.2 Data driven models

Data driven models have the advantage that they can be trained and maintained by machine learning experts independent of the language of the data. However, for annotation and maintenance of the training data native speakers of the respective language are necessary. If the annotation is kept plain and simple, basic linguistic skills are sufficient. This is the case for an annotation which consists of semantic concepts that are annotated per utterance [4].

Generative approaches based on Hidden Markov models or other approaches that utilize a hidden structure can align semantic concepts with a given word sequence [4][5][6][7][8][9]. Such an alignment is usually not necessary in a practical SDS, although it may be useful in the process of creating, correcting and maintaining the annotation of training and test corpora. Discriminative approaches create a feature vector from the utterance and classify it into a semantic category using standard classifiers such as conditional random fields [10][11] or support vector machines (SVM) [12][13][14][15]. A straight forward approach uses a multi-class SVM for dialog act classification and binary SVM for concept-value pair recognition. All results are combined to a semantic tuple classifier. The feature set contains word level unigrams, bigrams and trigrams [16]. Other approaches used linguistic features like parts of speech (POS) [17] or different kinds of kernels, like convolution [18] or string kernels [19].

2 Data for training and testing

We used a corpus that is specially annotated for spoken language understanding. It is described in [20] along with very good results using SVM. The corpus is available in the internet and contains 10571 transcribed utterances of human-machine conversations of a restaurant information system in English. The following annotations are provided [20]:

- Orthographic transcription
- Semantic annotation: dialog acts (e.g. inform), concept-value pairs (e.g. area=north)
- A list of the 10 best hypotheses generated by a speech recognizer (WER: 37%)
- Dialog act of previous system utterance

We cleaned up the orthographic transcriptions of both training and test set i.e., normalized spelling variants, corrected typing errors and normalized all characters with respect to case, resulting in training set vocabularies of 577 and 336 words for transcriptions and hypotheses respectively. In order to get an idea on how the performance of our SLU system depends on the word error rate (WER) of the used recognizer, we tested our models on the transcriptions (WER=0%) and on the best hypothesis of a recognition result as provided in the corpus (WER=37%).

3 Experimental setup

Speech recognizers usually provide the concept of grammar slots [21]. These are placeholders in an ASR-grammar which can be filled with lists of entries. We only treated semantic concepts *Food* and *Name* as ASR-grammar slots. In most recognizers the slot information is noted in the recognition result. We simulated this by collecting all instances of names and food types in the training set. We replaced all instances of this collection in both test and training set by the respective slot name. This means, that variants of instances that only occur in the test set could not be replaced. The second and third occurrences of the same slot type in one utterance are marked differently. The sentence "*i would like indian or italian food*" would be converted to something like "*i would like FOOD-1 or FOOD-2 food*". This simple transformation allows us to add new values without re-training the classifier. But it would also convert an *"italian sports car"* into a *Food* type, so we just use it for pre-processing.

We grouped the annotations such that we could train twelve different classifiers: *DA* (17 Dialog Acts), *Food* (6 classes), *Name* (4 classes), *Address* (3), *Area* (9), *Phone* (3), *Postcode* (3), *Pricerange* (8), *Signature* (2), *Task* (2), *Type* (2), and finally *Don't-care* (2). Exactly one of the 17 dialog acts is assigned to each utterance. All other classifiers are named according to the concept that they handle and have a class for each value that is possible with this concept. They also have a not-present class that was annotated when a certain concept was not in the

semantic annotation. An utterance which is annotated as *inform(food=italian, area=south)* would fall into the category not-present for the classifiers of *Name*, *Address*, *Phone*, *Pricerange*, *Postcode*, *Signature*, *Task*, *Type*, and *Don't care*. For all our experiments we used WEKA [22] as our test and training environment. As WEKA's built-in SVM implementation turned out to be a bit slow, we used LibSVM [23] via WEKA for training and testing where possible. For our classification results we used dialog act accuracy and F-score as described in [20]. *F'* is defined as the *harmonic* mean of precision and recall of the dialog act and all concept-value pairs of an utterance. *F* is calculated as the *arithmetic* mean over *F'* of all utterances in the test set. For this calculation we used the values as predicted by the classifiers. We did not attempt to correct the result in order to be compliant to the syntax of the semantic annotation, which has constraints on what concepts can co-occur with certain dialog acts. Annotations such as *inform(area)* or *request(area=east)* are not allowed by the syntax [20], but could occur in our classification results. Finally we are just interested in word based features, so we do not use knowledge about the previous system dialog act in our experiments.

4 Baseline system

We started with models that use a combination of uni- and bigrams as features, similar to [16] and [20], where uni-, bi- and trigrams were used. A feature is set to 1 if the word or bigram that corresponds to it is used in the sentence otherwise the feature remains 0. This leads to huge, but sparse vectors, which allow efficient calculations. We used linear kernel and optimized the parameters with grid search.

4.1 Reducing the imbalance of classes

A particular problem of our class design is that the *not-present* class is usually much bigger than the others. This may cause a classifier to be biased such that it assigns a data point more often to the *not-present* class just because it is so frequent. We tried to reduce the *not-present* class in the training sets. Table 1 shows results for models that use unigrams and bigrams as features. The models are trained on the *complete* data set and on a *reduced* data set, where we removed all sentences that are annotated with the *request* dialog act from the training data for the *not-present* classes of all concept classifiers.

Uni + bigram	Features	Test	DA Acc.	F ± 1.96 σ (%)	Size (kB)
complete trans.	4161	trans.	96.54	86.46 ± 0.5	14917
		1-best	71.94	64.78 ± 1.1	
complete 1-best	4050	trans.	82.89	86.35 ± 0.9	16048
		1-best	78.20	77.43 ± 1.0	
reduced trans.	4161	trans.	--	95.72 ± 0.4	13619
		1-best	--	73.73 ± 1.2	
reduced 1-best	4050	trans.	--	84.89 ± 0.9	15714
		1-best	--	76.15 ± 1.0	

Table 1 - Dialog act accuracy (Acc.) and F-score for classification on the transcriptions and the best hypothesis (1-best) of the test set. Classifiers are trained on both uni- and bigrams of the transcriptions (trans) and the best hypothesis (1-best) of the complete and reduced training data.

The models that were trained on the *reduced* transcriptions outperformed the models which were trained on the *complete* transcriptions, by roughly 10% absolute, when comparing *F* for both transcriptions and best ASR hypotheses. For the models trained on the hypotheses of the

ASR result (1-best) reducing the training data did not work at all. This means that training data reduction in order to improve the classifier must be done for each data type separately, if it works at all. For dialog act classification it does not make sense to remove the *requests* from the training data, as the *requests* are to be predicted by the model. *F* considers the dialog act and all concepts. In all further experiments we used the *reduced* data for concept classifiers trained on transcriptions.

4.2 Limited benefit with bigram features

For this experiment, we trained two new model sets, one on the reduced transcriptions and the other on the complete set of best hypothesis of the recognition results of the training set. For both we used just unigrams as features. When moving from uni- and bigrams (Table 1) to just unigrams (Table 2), the feature space shrinks by a factor of 7 and 12 respectively. This causes a decrease in the model sizes by a factor of 5. For the models trained on orthographic transcriptions, the results on the expert user task showed only very limited loss. We observed the biggest degradation for the dialog act classifier with 0.27% absolute. The *F* difference is mainly caused by the dialog acts. For the concept-value pair classifiers the results decreased marginally, in two cases they even improve. The results for the ASR hypotheses are similar, just a bit lower. For the models trained on the best hypotheses, we observed considerable improvements everywhere apart from the dialog tested on ASR hypotheses. In summary, it seems that unigrams already contain a lot of information. Adding bigrams yields only small gain.

Unigram	Features	Test	DA Acc.	F ± 1.96 σ (%)	Size (kB)
reduced trans.	577	trans.	**96.27**	**95.45 ± 0.4**	2791
		1-best	71.49	73.35 ± 1.2	
complete 1-best	336	trans.	85.62	88.45 ± 0.8	3954
		1-best	**77.98**	**77.43 ± 1.0**	

Table 2 - Dialog act accuracy (Acc.) and F-score of the classification results tested on the transcriptions and the best hypotheses of the test set. Classifiers are trained on just unigrams of the transcriptions and the best hypothesis (1-best) of the training data.

5 Feature reduction

5.1 Drawbacks of LDA and PCA

Obvious candidates would be Principal Component Analysis (PCA) [24] and Linear Discriminant Analysis (LDA) [25]. These learn a mapping from the original feature space to a new feature space in which the features are ordered according to their importance. This ordering could be used to reduce the new feature space but the original feature space would remain the same and the mapping would be an extra resource that must be stored on the device or held in memory in addition to the SVM models. Furthermore the software on the device would then consist of the feature space mapping algorithm and the SVM decoder. As the new features are likely not to be sparse integer vectors anymore, we expect to get rounding errors and higher computational effort. These concerns made us search for alternative solutions.

5.2 The SVM RFE algorithm

The literature about Recursive Feature Elimination with SVM (SVM RFE) [26] reported impressive results on selecting genes responsible for cancer from gene sets of healthy and

cancerous tissue. Furthermore this method does not have the above mentioned drawbacks. Features are directly removed from the original feature space. The algorithm is tailored to SVM. No additional mapping algorithms are necessary on the device, only the training gets more time consuming, which is acceptable. The SVM RFE algorithm uses an SVM to create a list of all features ranked by their importance for the classification task. In the following, we shortly explain the SVM RFE algorithm for a binary classification task as it is described in [26] and implemented in WEKA:

1. An SVM with a linear kernel is trained on the entire training data. The result is a weight vector $\boldsymbol{w}$, which is perpendicular to the decision hyperplane. Each component w_i of this vector corresponds to one dimension of the feature space. The value of w_i^2 indicates the importance of the corresponding feature for the classification.
2. All w_i^2 are calculated for the current feature set.
3. The feature that corresponds to the lowest w_i^2 is dropped from the original feature set and would be entered on top position of the current ranked list.
4. The SVM is trained with the remaining feature set.
5. If (size of feature set > 1) then go to 2.

The ranked list contains all features. The top ranked entry is the most important feature as it is the last that was added to the list and the bottom ranked feature is the least important as it is the first that was added

5.3 Feature selection for concepts

As it turns out, the number of features and the size of our training data are quite challenging for the implementation of SVM RFE in WEKA, which we used in our experiments. We therefore decided to start the feature selection with 577 unigram features and trained the models on the reduced transcriptions of the training set. As an example we show the results for the area classifier, that has nine classes:

area=centre, area=south, area=west, area=east, area=north, *area≠east, area=dontcare*

area (requested) and *area-not-present* (not annotated with any of the *area* values)

The selection algorithm returned these top ranked features: *riverside, south, west, east, north, anywhere, near, southern, recorded, centre, any, northern, right, peats, you, note, in, central, path, care, part, after, i, stuff, eastern, area, moderately, the, love, or, recommend, priced,…*

Many of these features make perfect sense. Those that seem unrelated to *area* may be useful to distinguish the *area-not-present* class from all others. Starting with the top 5 features, we successively selected more features from the ranked list, built models for each new feature set on the training data and tested on the test data. Figure 1 shows accuracy plots for some of the concept classifiers (trained and tested on transcriptions). The classifiers for *Pricerange* show a behavior as we expected it: The accuracy increases with a growing number of features. We observe a steep increase for small feature sets. For large feature sets the accuracy saturates, reaching the best value for the full unigram set, that has 577 features. All other classifiers reach their optimum with considerably smaller feature sets. The classifiers for *Phone*, *Signature*, and *Postcode* reach the best accuracy already with 5 features which seems plausible as they have just 2 or 3 classes. The classifiers for *Address*, *Name*, *Area*, *Type*, and *Don't Care* (some are not displayed) reach the best accuracy with a feature set somewhere between 11 and 32 features. The *Food* classifier reaches the best accuracy with 57 features.

We tested the same models on the best hypotheses of the recognition results. The accuracies are lower of course, but the general picture of the graphs is similar. Even 5 classifiers reach

the best accuracy already with 5 features. Just the *Pricerange* classifier reaches the accuracy maximum only with the full set of unigrams.

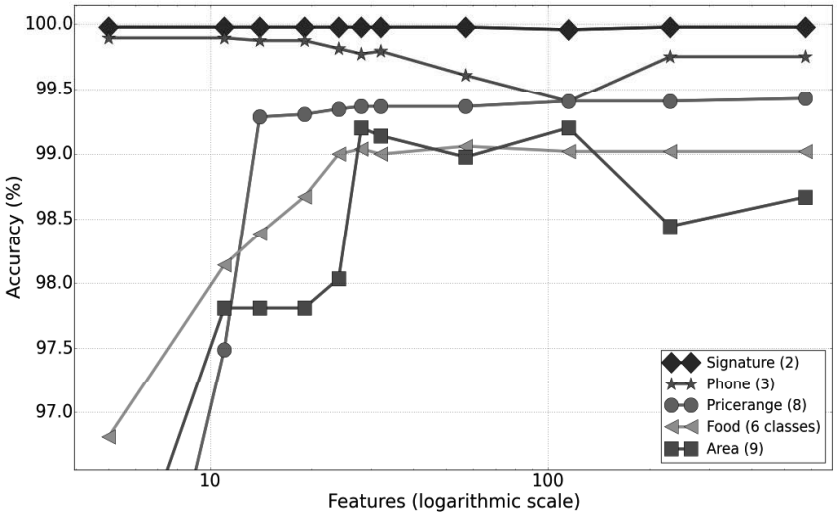

Figure 1 - Concept classifiers trained on successively reduced feature sets. Trained and tested on transcriptions.

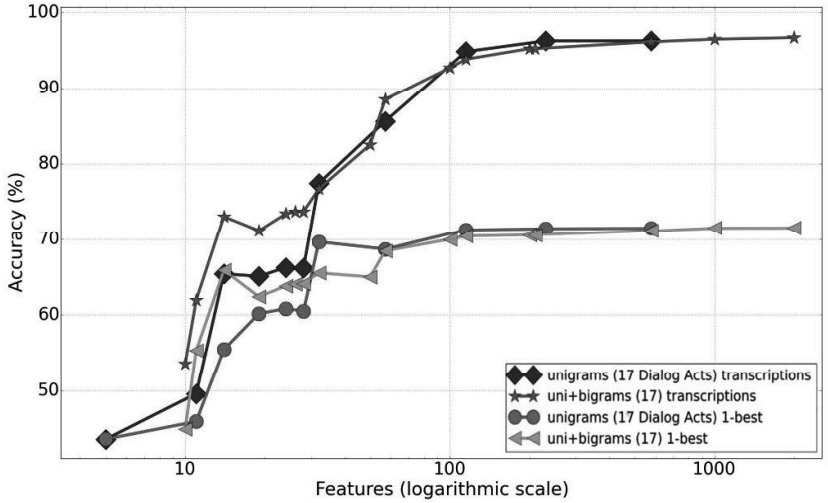

Figure 2 - Dialog act classifier trained on successively reduced feature sets. Trained and tested on transcriptions.

5.4 Feature selection for dialog acts

For dialog acts we did feature selection on two feature sets. In the first experiment we used a set of 577 unigrams and applied the algorithm on the transcriptions of the complete training

set. The second experiment was conducted with a feature set of 4161 uni- and bigrams. The number of features combined with the size of the training set was too big to get processed in reasonable time. We therefore developed a faster algorithm: The training set was divided into 5 parts. We prepared a ranking for each partial data set. To combine the resulting 5 rankings, we selected from each partial ranking the top R features, combined them to a common list and finally kept all features that occurred more than three times in the list (majority vote). Raising the value of R = 1, 2, 3, … allows selecting a successively increasing number of features.

In two experiments, we trained models with different feature sets and tested them on both the transcriptions and the best hypothesis of the recognition result. The accuracies of these models are displayed in Figure 2. Obviously all models perform better on the transcriptions than on the recognition results. In both use cases the bigram model is considerably better for models with less than 32 features. Around that point the unigram models improve substantially and overtake the bigram models. For larger feature sets both model types saturate. With feature sets larger than the vocabulary size of 577, the bigram models can slightly improve over the best unigram with 230 (trans.) or 577 (1-best) Features. The best 26 features of the uni- and bigram model are *(</s>* for sentence end): *where-it, right, hello, thanks-</s>, no, zippier-</s>, bye, what-types, your-thank, hi, goodbye, is-just, yes, the, phone-phone, your-phone, post-codes, yeah, is-the, yes-i, what, the-place, your-list, thank-</s>, else, what-price*

5.5 Best selection

We selected the best concept-classifiers amongst those with 32 or fewer features and the uni- and bigram dialog act classifier with 200 features. This combined model uses 764kB and achieves $F = 95.56 \pm 0.48$ when tested on transcriptions.

6 Conclusions

On the used corpus, adding bigrams to the feature space of concept-value pair SVM-classifiers leads to only small gains. The results of the feature reduction experiments showed that 9 out of 11 concept classifiers reach their best performance with a feature set of 58 or fewer words (10% of vocabulary). Many of the other features seem to rather introduce noise than help with the classification. The SVM classifier copes well with these features, as the accuracy differences between the best model and the model trained with the full feature set are smaller than 0.2% for all except for the *Area* classifier. For the dialog acts we observe a different behavior. More features result in better models and bigrams bring a performance gain that gets smaller for large feature sets. We successfully reduced the SVM model size by a factor of 17 while maintaining good classification performance. Combining all results yields small and performant models for SLU which are well suited for embedded devices.

7 References

[1] G. Tur and R. De Mori (eds), Spoken Language Understanding: Systems for Extracting Semantic Information from Speech, New York: John Wiley & Sons, 2011.

[2] W. Ward, "Extracting information in spontaneous speech", ICSLP 1994 – The 3rd International Conference on Spoken Language Processing, Yokohama, Japan, 1994.

[3] S. Seneff, "TINA: Natural language system for spoken language applications", Computational Linguistics, vol. 18, no, 1, pp 61-86, 1992.

[4] R. Pieraccini and E. Levin, "Stochastic representation of semantic structure for speech understanding", 2nd Eurospeech, Genova, Italy, 1991.

[5] R. Schwarz, S. Miller, D. Stallard, and J. Makhoul, „Language understanding using hidden understanding models", ICSLP 1996 – The 4th International Conference on

Spoken Language Processing, Philadelphia, PA, USA, 1996.
[6] Y. He and S. Young, "Semantic processing using the hidden vector state model", Computer Speech & Language, vol. 19, no. 1, pp. 85-106, 2005.
[7] Y. He and S. Young, "Spoken language understanding using the hidden vector state model", Speech Communication, vol. 48, no. 3-4, pp. 262-275, 2006.
[8] A. Acero and Y. Y. Wang, "Spoken language understanding", IEEE Signal Processing Magazine, vol. 22, no. 5, Sept. 2005.
[9] I. V. Meza-Ruiz, S. Riedel, and O. Lemon, "Accurate statistical spoken language understanding from limited development resources", ICASSP 2008 – IEEE International Conference on Acoustics, Speech and Signal Processing, 2008.
[10] Y. Y. Wang and A. Acero, "Discriminative models for spoken language understanding", INTERSPEECH 2006 – The 9th International Conference on Spoken Language Processing, Pittsburgh, PA, USA, 2006.
[11] D. Zhou and Y. He, "Learning conditional random fields from unaligned data for natural language understanding", Advances in Information Retrieval, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, pp. 283–288, 2011.
[12] V.N. Vapnik, Statistical learning theory, New York: John Wiley & Sons, 1998.
[13] B. Boser, I. Guyon, and V. Vapnik, "A training algorithm for optimal margin classifiers", Proceedings of the workshop on Computational Learning Theory, Pittsburgh, PA, USA, pp. 144-152, 1992.
[14] J. C. Platt, "Sequential minimal optimization: A fast algorithm for training support vector machines", Microsoft Research, TechReport MSR-TR-98-14, 1998.
[15] N. Christianini, and J. Shawe-Taylor, An introduction to support vector machines, Cambridge University Press, Cambridge, MA, 1999.
[16] F. Mairesse, M. Gašić, F. Jurčíček, S. Keizer, B. Thomson, K. Yu, and S. Young, "Spoken language understanding from unaligned data using discriminative classification models", ICASSP 2009 – IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 4749-4752, 2009.
[17] S. Pradhan, W. Ward, K. Hacioglu, J. Martin, and D. Jurafsky, "Shallow semantic parsing using support vector machines", Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL: HLT-NAACL, 2004.
[18] A. Moschitti, "A study on convolution kernels for shallow semantic parsing", Proceedings of the 42nd Annual Meeting of the ACL, 2004.
[19] R. J. Kate and R. J. Mooney, "Using string-kernels for learning semantic parsers", Proc. of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL, 2006.
[20] M. Henderson, M. Gašic, B. Thomson, P. Tsiakoulis, K. Yu, and S. Young, "Discriminative spoken language understanding using word confusion networks", SLT 2012 – 4th IEEE Workshop on Spoken Language Technology, Miami, FL, USA, 2012.
[21] M. C. McCord, "Slot Grammars", American Journal of Computational Linguistics, vol. 6, no. 1, 1980.
[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, "The WEKA Data Mining Software: An Update" SIGKDD Explorations, vol. 11, no. 1, 2009.
[23] C.-C. Chang and C.-J. Lin, "LIBSVM : a library for support vector machines", ACM Transactions on Intelligent Systems and Technology, vol. 2, no. 27, pp. 1-27, 2011.
[24] K. Pearson, "On lines and planes of closest fit to systems of points in space", Philosophical Magazine 2:559-572, 1901.
[25] R. A. Fisher, "The use of multiple measurements in taxonomic problems", Annals of Eugenics, vol. 7, pp. 179-188, 1936.
[26] I. Guyon, J.Weston, S.Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines", Machine Learning, vol. 46, pp.389-422, 2002.