A COMPARATIVE STUDY OF RECURRENT NEURAL NETWORK MODELS FOR LEXICAL DOMAIN CLASSIFICATION

Suman Ravuri[1,3] Andreas Stolcke[2,1]

[1]International Computer Science Institute
[2]Microsoft Research, Mountain View, CA, USA
[3]University of California, Berkeley, CA, USA
ravuri@icsi.berkeley.edu anstolck@microsoft.com

ABSTRACT

Domain classification is a critical pre-processing step for many speech understanding and dialog systems, as it allows for certain types of utterances to be routed to specialized subsystems. In previous work, we explored various neural network (NN) architectures for binary utterance classification based on lexical features, and found that they improved upon more traditional statistical baselines. In this paper we generalize to an n-way classification task, and test the best-performing NN architectures on a large, real-world dataset from the Cortana personal assistant application. As in the earlier work, we find that recurrent NNs with gated memory units (LSTM and GRU) perform best, beating out state-of-the-art baseline systems based on language models or boosting classifiers. NN classifiers can still benefit from combining their posterior class estimates with traditional language model likelihood ratios, via a logistic regression combiner. We also investigate whether it is better to use an ensemble of binary classifiers or a NN trained for n-way classification, and how each approach performs in combination with the baseline classifiers. The best overall results are obtained by first combining an ensemble of binary GRU-NN classifiers with LM likelihood ratios, followed by picking the highest class posterior estimate.

Index Terms— Domain classification, neural networks, LSTM, GRU, recurrent networks.

1. INTRODUCTION

Utterance classification is an important pre-processing step for many dialog systems that interpret speech input. For example, a user asking Siri or Cortana to "tell me about the weather" should have her utterance classified as *weather-query* so that the query can be routed to the correct natural understanding subsystem. In previous work [1], we compared feedforward neural network addressee models in a related task of lexical addressee detection, in which a system must identify whether speech is directed at the machine, or another human, and recently, we compared recurrent neural network (RNN) and long short-term memory (LSTM) units [2] for both addressee and intent detection. In [3], we compared gated recurrent networks (GRUs), LSTMs, RNNs, and feedforward neural network addressee models on both small and large corpora, but again using only binary domain/intent classification tasks.

The motivation for neural network models is that previous n-gram-based classification approaches, such as standard LMs and boosting, suffer from two fundamental and competing problems: the limited temporal scope of n-grams, and their sparseness, requiring large amounts of training data for good generalization. The longer the n-grams one chooses to model, the more the sparseness issue is exacerbated. Neural network models can use continuous word embeddings to help generalizing unseen word combination from those of similarly distributed words, based on the principle that similar words are represented by nearby vectors. Furthermore, recurrent NNs can learn how much word history to represent instead of being limited by the fixed length of n-grams. can be improved by enlisting out-of-domain data to train word embeddings [1].

In this study we take the best-performing approaches from prior work on binary domain and intent classification [3] and generalize them to n-way classification, which represents the most common application scenario. The most effective approaches, based on previous work, involved recurrent neural network sequence classifiers, in combination with traditional language models. As we generalize these techniques to n-way classification, several choices for the setup of the classifiers and their combination arise, and will be compared. We focus our investigation on a large Cortana dataset, i.e., a real-world classification task that is key to today's conversational personal assistants.

There is a vast literature on domain and intent classification for purposes of speech understanding; for prior work see [4, 5] and references therein.

2. COMPARISON SYSTEMS

2.1. Baseline systems

As our experimental baseline we use two classifier architectures based on n-gram features. One system is a pair of class-specific n-gram language models, each of which computes a class likelihood. The log ratio of these likelihoods is then normalized for the utterance length (number of words) to obtain a detection score that is thresholded (for binary classification) or from which posterior probabilities may be estimated (via logistic regression, see below). A detailed study of this approach can be found in [6].

The other baseline approach is the "Boostexter" boosting algorithm [7, 8]. Boosting generates output scores that are not inherently probabilistic in nature, but may be transformed (via logistic regression) to posterior estimates as well.

2.2. RNN-based utterance classifier

Recurrent neural network language modeling (RNNLM) [9] grew out of the observation that temporal modeling of a sentence at the hidden layer can outperform models based on the Markov (limited memory) assumption, since the former kind of modeling can in the-

ory store relevant information for many more time steps. Similar to Neural Network Language Model [10], the RNNLM maps words to a dense n-dimensional word embedding, but unlike the feedforward NNLM, the hidden state h_t is a function of the current embedding and the previous hidden state (and a bias):

$$h_t = \sigma(W_t h_{t-1} + v_t + b_h)$$

Moreover, empirically, we found in previous work that the optimal dimension for the word embedding is less than half of that in the feedforward language model, leading to more compact models.

Adapted for use in a binary utterance classification task, we can train a single RNN model on utterance class labels, shown in Figure 1. The RNN attempts to classify the utterance based on the information stored thus far in h_t. At test time, the probability of an utterance label is calculated as:

$$\begin{aligned} P(L|\mathbf{w}) &\approx P(L_1, \ldots, L_n|\mathbf{w}) \\ &= \prod_{i=1}^{n} P(L_i|\mathbf{w}) \approx \prod_{i=1}^{n} P(L_i|w_i, h_{i-1}) \\ &= \prod_{i=1}^{n} P(L_i|h_i) = \prod_{i=1}^{n} softmax(W_o h_i + b_o) \end{aligned}$$

where the final equality is embodied in the softmax output function.

The naive multiclass extension to these models would be to predict one of n labels instead of of 2. For the RNN model, however, such an approach yields very poor results. Instead, we train n separate binary classifiers, in which the model attempts to classify whether or not an utterance belongs to a particular class. Then a separate logistic regression model uses the n output classification scores as input and predicts one of n labels. One issue we encountered in the corpus studied here is that for some labels, the number of negative examples far outnumbered the number of positive ones, and training on the entire dataset would lead to unbalanced priors. In order to sidestep this problem, we subsample more numerous utterances so that the ratio of positive to negative examples is 1:1. While at first glance, the binary approach seems unsatisfying because n copies of the model need to be trained, with subsampling training can be parallelized trivially, and moreover, each model can be trained much more quickly. A downside of this method, however, is that inference requires n predictions, and unless the number of classes to be predicted is very large (in which case models using "naive" approach would spend most of its time in the output layer), then the amount of computation is roughly n times as much as the "naive" approach. In Section 4, this combination of n binary classifiers is denoted as **"binary."**

2.3. LSTM- and GRU-based utterance classifier

Ideally a model performing utterance classification would predict a single class label per utterance. In earlier work, we did not obtain competitive performance with RNN models predicting a single label at the end of an utterance even for a two-class task. We surmise that the poor performance stemmed from the vanishing gradient problem. This suggests the use of long short-term memory (LSTM) units for utterance classification.

The LSTM, first described in [11], attempts to circumvent the vanishing gradient problem by separating the memory and output representation, and having each dimension of the current memory unit depending linearly on the memory unit of the previous timestep. A popular modification of the LSTM uses three gates—input, forget, and output—to modulate how much of the current, the previous, and output representation should be included in the current timestep. Mathematically, it is specified by the equations:

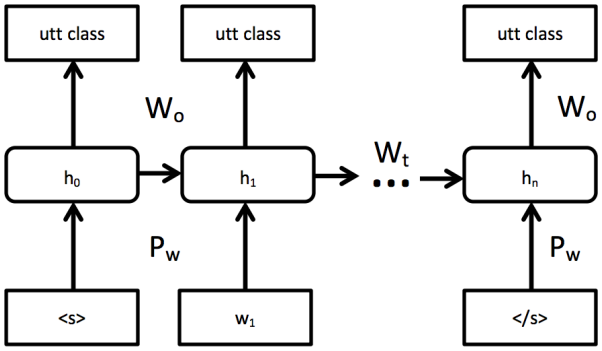

Fig. 1. *RNN utterance classifier model.*

$$\begin{aligned} i_t &= \sigma(W_i v_t + U_i h_{t-1} + b_i) \\ f_t &= \sigma(W_f v_t + U_f h_{t-1} + b_f) \\ o_t &= \sigma(W_o v_t + U_o h_{t-1} + V_o m_t + b_o) \\ m_t &= i_t \circ tanh(W_c v_t + U_c h_{t-1} + b_c) + f_t \circ m_{t-1} \\ h_t &= o_t \circ tanh(m_t) \\ P(L|\mathbf{w}) &= softmax(W_o h_T + b_o) \end{aligned}$$

where i_t, f_t, and o_t denote the input, forget, and output gates respectively, m_t, the memory unit, and h_t, the hidden state, and is shown in Figure 2. We found that, unlike for RNNs, a model making a single prediction at utterance end does achieve good performance, and outperforms models that make predictions after every word.

More recently, gated recurrent units have been proposed ([12]) as a simplification of the LSTM, while keeping the ability to retain information over long sequences. As in the LSTM, "memory" is handled by a simple linear interpolation between a hidden-like state in the previous time step and a RNN-like component representing a current time step. Unlike the LSTM, however, it uses only two gates, memory units do not exist, and the linear interpolation occurs in the hidden state. We use a slight modification of the original GRU, proposed in [13], as is described by equations:

$$\begin{aligned} z_t &= \sigma(W_z v_t + U_z h_{t-1} + b_z) \\ r_t &= \sigma(W_r v_t + U_r h_{t-1} + b_r) \\ h_t &= z_t \circ tanh(W_h v_t + U_h(r_t \circ h_{t-1})) + (1 - z_t) \circ h_{t-1} \\ P(L|\mathbf{w}) &= softmax(W_o h_T + b_o) \end{aligned}$$

One question is whether the one-hot vector w should input directly to the LSTM or GRU, as proposed by [14] for a slot-filling task. We found it better to use a separate linear embedding, and use the embedding v_t as input to the LSTM or GRU. Figure 2 depicts this model.

For the gated models, using the naive approach yields good results, we also compare this method to the binary approach. In Section 4, the naive approach is denoted as **"multiclass."**

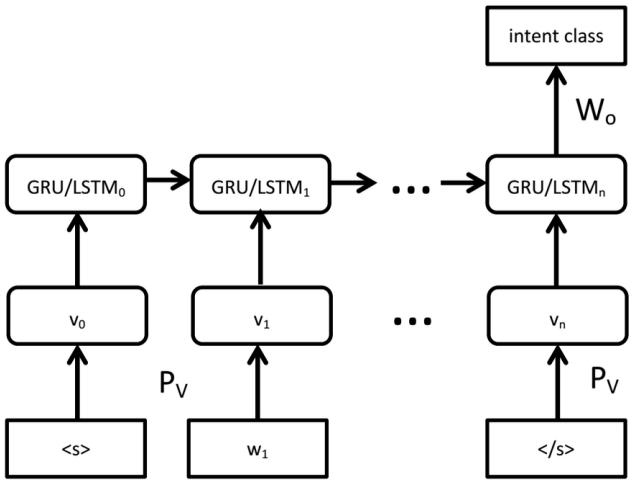

Fig. 2. *LSTM/GRU utterance classifier model.*

3. METHOD

3.1. Cortana Domain Classification

The corpus is drawn from the Microsoft Cortana personal assistant [4], which consists of user input to the system extracted from a corpus of real interactions. Utterances directed at the system need to be routed to one of nine semantic subsystems based on the domain of discourse (such as communication, weather, etc.), and those for which no specialized handling is available are treated as web search queries. For this task, the model must correctly classify one of these nine classes. A corpus of 2.1 million utterances comprises the training set. Temporally later and disjoint utterance sets were used for development/tuning (138k utterances) and testing (221k utterances). The utterances for all three sets are a mix of one-best outputs from an automatic speech recognition system and typed user commands.

3.2. Experimental Setup

The recurrent neural network, long short term memory, and gated recurrent unit models use a 200-dimensional word embeddings, as those parameters experimentally produced the best results. In addition, the LSTM and GRU included a layer of 15-dimensional hidden and, in the case of the LSTM, memory units. Not including word embeddings, the LSTM model has roughly 150% more parameters than the RNN, while the GRU uses fewer than the RNN. For small-vocabulary tasks, this choice in neural architecture can lead to LSTM models being substantially larger than the RNN, but for large-vocabulary tasks such as the one studied here, the size of the embeddings dwarf the number of other parameters, so we just use the structure which produced the best results.

3.3. Training and Evaluation

Authors of other works have noted that parameter estimation for RNNs is substantially more difficult than for feedforward networks. Well-trained systems typically use a combination of momentum, truncated back-propagation through time (BPTT), regularization, and gradient clipping. Since utterance lengths for the corpora investigated were typically under 20, we found no improvement employing gradient clipping or truncated BPTT. Moreover, regularization had either minimal or deleterious effect. In previous work on smaller corpora, we found momentum to help, but for the Cortana corpus, momentum – neither simple nor more advanced modifications such as Nesterov momentum [15] – did not yield better results. Learning followed a slight modification to the newbob training schedule: the learning rate is decreased by half when the cross-entropy on a held-out set improve by less than 0.1% twice, and after the second decrease, the learning rate was halved in each epoch until cross-entropy on the heldout set did not decrease. The initial learning rate for the RNN models are 1.0, while for the gated models they were 0.05. The neural network models were trained on 90% of the training set, while the heldout set is the remaining 10%.

3.4. Binary versus n-way classification

The boosting, RNN, LSTM, and GRU classifiers can naturally perform n-way classification; the NN classifiers all output class posterior estimates. All systems can also be used as binary classifiers, in which case we construct an n-way classifier as follows: the output scores for a class and for its complement are input to a linear logistic regression (LLR), and trained on the development set. The LLR output is a posterior probability estimate for its target class. The class with highest posterior estimate is the output of the n-way classifier.

LLR can also be used to combine multiple estimators at the score level [16]. When combining two sets of binary classifiers, we train LLR using output scores of both types of classifiers as input. Again we obtain n posterior estimates, and pick the largest.

For combining two n-way classifiers we simply average their class-wise posterior estimates, and pick the class with the highest combined posterior.

3.5. Performance metrics

We evaluate all system using two metrics. First is the overall classification accuracy, which is also the micro-average of the class-wise recall rates. Second, we compute F1 scores (harmonic means of recall and precision) for each utterance class, and macro-average these into a single aggregate F1 score.

4. RESULTS AND DISCUSSION

Table 1 summarizes the results of both single-model and model combination experiment. Looking first at baseline systems, we note that the ensemble of binary boosting classifiers outperforms the ngram LM system. We found that the language model benefits from word 4-grams over 3-grams, whereas the boosting classifier did not (and we therefore used up to trigram features only). Also, we found that n-way boosting was not competitive with the ensemble of binary boosting classifiers (and is not reported here).

Turning to the neural network models, we found that the RNN model does not outperform either baseline, a multiclass version of the RNN was not competitive (not reported in the table). By contrast, both the LSTM and GRU outperform both baselines on accuracy, and are competitive with the boosting baseline on F1 measure. Results for binary versus multiclass models depend on the performance metric: binary classifiers are superior in terms of F1 measure, but multiclass models are better in terms of accuracy. LSTMs and GRUs offer roughly similar performance.

For the combination systems, even the recurrent neural network model improved results, accuracy increased by 1.1% and F1 increased by .012 over the word 4-gram baseline. That said, the RNN

Table 1. *Cortana domain classification results. The top portion shows results for the baseline systems, the second-from-top results for the neural network models, the second-from-bottom combination results using a linear logistic regression combiner, and bottom combination results using posterior averaging. For baseline systems note that boosting with 4-grams performed worse than with trigrams, so only the trigram result is reported here.*

System	Accuracy (%)	F1
word 4-gram LM-binary	90.20	.8883
word trigram boosting LM-binary	91.03	.9028
RNN-binary	89.31	.8774
LSTM-binary	91.08	.9002
LSTM-multiclass	**91.32**	.8986
GRU-binary	91.13	**.9006**
GRU-multiclass	91.29	.8989
word 4-gram+RNN-binary	91.37	.9032
word 4-gram+LSTM-binary	91.84	.9101
word 4-gram+GRU-binary	**91.90**	**.9104**
word trigram boosting+LSTM-binary	91.77	.9091
word trigram boosting+GRU-binary	91.78	.9091
word 4-gram+LSTM-multiclass	91.82	.9097
word 4-gram+GRU-multiclass	91.82	.9100

model is uncompetitive with the gated networks, as those systems were roughly 0.5% better in accuracy and .07 better in F1 measure. Interestingly, combining with word 4-grams was better than combining with boosted word trigrams, even though by itself the boosting classifier outperforms the 4-gram language model. A possible reason for this behavior is that the language model classifiers outputs log probability scores, whereas the boosting classifiers do not.

Finally, posterior averaging of either the LSTM-multiclass or GRU-multiclass systems with the word 4-gram systems is almost as good as linear logistic regression combination of the word 4-gram systems with the best binary neural network models. Therefore, if simplicity of the overall system is a concern the multiclass versions of the gated – the long short term memory and gated recurrent units – networks are a good choice.

5. CONCLUSIONS

We have compared traditional and neural word sequence classifiers on the task of domain classification, using a large real-world corpus (Cortana user input), while also considering ensembles of binary versus n-way classification architectures. The results are largely in line with prior results on binary utterance classification, with gated unit recurrent networks – LSTM and GRU – performing best, and beating (at least in overall accuracy) the baseline systems. The most effective overall classifiers involved ensembles of binary classifiers that were each combined with a corresponding ngram language model classifier via logistic regression, giving about 17% relative error reduction over the ngram LM by itself, or 10% over the boosting classifier. The results suggest that if one were to choose a neural network model, the GRU seems the superior choice, as it offers the best results, or results competitive with the LSTM while also being a somewhat simpler model.

For future work, we should note that one limitation of the present study is that we only examine lexical information, and what can be inferred from the utterance at hand. In future work it would worthwhile to incorporate nonlexical (e.g., prosodic) information [17], as well as utterance context preceding the one to be classified [4]. With regard to modeling, the inclusion of a layer performing convolution on the word sequence [4] is a promising architectural feature that is orthogonal to the aspects studied here. Finally, we would like to consider word representations that more directly model sub-word units, as a current limitation of the word embedding approach is that words such as "carrot" and "carrots", which are very similar, do not share any statistical strength.

6. ACKNOWLEDGMENTS

We wish to thank Minwoo Jeong and Ruhi Sarikaya for assistance with the Cortana data, and Kaisheng Yao for helpful discussions on LSTM models.

7. REFERENCES

[1] Suman Ravuri and Andreas Stolcke, "Neural network models for lexical addressee detection," in *Proc. Interspeech*, Singapore, Sept. 2014, pp. 298–302.

[2] Suman Ravuri and Andreas Stolcke, "Recurrent neural network and LSTM models for lexical utterance classification," in *Proc. Interspeech*, Dresden, Sept. 2015.

[3] Suman Ravuri and Andreas Stolcke, "A comparative study of neural network models for lexical intent classification," in *Proceedings IEEE Workshop on Automatic Speech Recognition and Understanding*, Scottsdale, Arizona, Dec. 2015.

[4] Puyang Xu and Ruhi Sarikaya, "Contextual domain classification in spoken language understanding systems using recurrent neural network," in *Proc. ICASSP*, Florence, May 2014, pp. 136–140.

[5] Gokhan Tur, Dilek Hakkani-Tür, Larry Heck, and S. Parthasarathy, "Sentence simplification for spoken language understanding," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*. May 2011, IEEE SPS.

[6] Heeyoung Lee, Andreas Stolcke, and Elizabeth Shriberg, "Using out-of-domain data for lexical addressee detection in human-human-computer dialog," in *Proceedings North American ACL/Human Language Technology Conference*, Atlanta, GA, June 2013, pp. 221–229.

[7] Robert E. Schapire and Yoram Singer, "Boostexter: A boosting-based system for text categorization," *Machine Learning*, vol. 39, no. 2/3, pp. 135–168, 2000.

[8] Benoit Favre, Dilek Hakkani-Tür, and Sébastien Cuendet, "icsiboost. open-source implementation of Boostexter," http://code.google.come/p/icsiboost, 2007.

[9] Tomáš Mikolov, Martin Karafiát, Lukaš Burget, Jan "Honza" Černocký, and Sanjeev Khudanpur, "Recurrent neural network based language model," in *Proc. Interspeech*, Makuhari, Japan, Sept. 2010, pp. 1045–1048.

[10] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," Tech. Rep. 1178, Department of Computer Science and Operations Research, Centre de Recherche Mathématiques, University of Montreal, Montreal, 2000.

[11] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[12] KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *CoRR*, vol. abs/1409.1259, 2014.

[13] Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014.

[14] Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi, "Spoken language understanding using long short-term memory neural networks," in *IEEE SLT*, 2014.

[15] Yu Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence $o(1/k^2)$," *Doklady AN SSSR (Soviet. Math. Docl.)*, vol. 269, pp. 543–547, 1983.

[16] Stéphane Pigeon, Pascal Druyts, and Patrick Verlinde, "Applying logistic regression to the fusion of the NIST'99 1-speaker submissions," *Digital Signal Processing*, vol. 10, no. 1-3, pp. 237–248, Jan. 2000.

[17] Elizabeth Shriberg, Andreas Stolcke, and Suman Ravuri, "Addressee detection for dialog systems using temporal and spectral dimensions of speaking style," in *Proc. Interspeech*, Lyon, Aug. 2013, pp. 2559–2563.