Neural Network Models for Lexical Addressee Detection

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

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

Abstract

Addressee detection for dialog systems aims to detect which utterances are directed at the system, as opposed to someone else. An important means for classification is the lexical content of the utterance, and N-gram models have been shown to be effective for this task. In this paper we investigate whether neural networks can enhance lexical addressee detection, using data from a human-human-computer dialog system. Even though we find no improvement from simply replacing the standard N-gram LM with a neural-network LM as class likelihood estimators, improved classification accuracy can be obtained from a modified neural net model that learns distributed word representations in a first training phase, and is trained on the utterance classification task in a second phase. We obtain additional gains by combining the class likelihood estimation and classification training criteria in the second phase, and by combining multiple model architectures at the score level. Overall, we achieve over 2% absolute reduction in equal error rate over the N-gram model baseline of 27%.

Index Terms: addressee detection, neural net language model, distributed word representations, multitask training.

1. Introduction

Addressee detection is a serious problem for dialog systems that operate in the presence of multiple humans, regardless of whether those humans are all interacting with the system or not, or even speaking at all. Users in the company of other humans tend to talk to others about what they are doing, to solicit input, or to have side-conversations (self-talk can also occur). The system therefore has to classify incoming speech to detect whether the utterances is meant for itself. In past work we have studied addressee detection leveraging several speech-based knowledge sources, such as what was said (lexical content) and how it was said (speaking style) [1]. Other researchers have also brought multi-modal cues to bear on this task [2, 3, 4, 5].

In this work we focus on speech-based addressee detection, and specifically on the modeling of lexical utterance content, for which statistical language models have proven to be an effective tool. Recent progress in language modeling based on neural-net-induced distributed word representations [6] raises the question whether such models can also improve addressee detection. In the remainder of this paper we try to answer this question, starting from a simple drop-in replacement of neural net languguage models (NNLMs) for the standard N-gram LMs, and then progressing to modified architectures that are tailored to the task at hand, and that combine word prediction with utterance classification. A key insight is that the training of word embeddings can be decoupled from the training of the utterance classifier. This also allows out-of-domain data, which is typically plentiful compared to data from the target domain, to be leveraged for training better NNLM models.

2. Method

2.1. Data

Data come from interactions between two acquaintances and a "Conversational Browser" (CB) dialog system using only spoken input. Subjects were brought into a room and seated about 5 feet away from a large TV screen and roughly 3 feet away from each other. They were told about the basic capabilities of the CB system and the domains it could handle, and were given a small set of short commands, such as to start a new interaction, pause, stop listening, or wake up the system. Other than that, subjects were told to use unrestricted natural language. The system detects starts and ends of utterances automatically. In this collection users did not use other modalities to indicate speech activity. More information about the dialog system itself and its spoken language understanding approach can be found in [7].

The resulting corpus comprises 6.3 hours of recordings over 38 sessions with 2 speakers each from a set of 36 unique speakers. Session durations ranged from 5 to 40 minutes, as determined by users. Speech was captured by a Kinect microphone-array; endpointing and recognition used an off-the-shelf recognizer. Although the full interaction was recorded, we used only the speech segments detected and recognized by the system; we simply call these "utterances."

A total of 6,920 segments from 38 sessions were hand-transcribed at the word level for evaluation and labeling purposes. The segments were then annotated for addressee by an experienced experimenter who had run subjects in the data collection. The experimenter had access to the audio and video recordings and reported that annotation was generally straightforward. After eliminating utterances containing no intelligible foreground speech the dataset consisted of 5,488 utterances, totaling 5.33 hours. Computer-addressed segments were also labeled by the annotator as either command or noncommand. Segments containing both human- and computer-addressed speech (in any sequence) were marked as mixed; since these were also processed by the system they were grouped with the computer-addressed class for detection purposes. The 38 available sessions varied greatly in length; the 22 shortest sessions were placed in the test set to maximize speaker and session variation. Table 1 gives the distribution of utterance types, and examples for each type.

For experiments with out-of-domain training data we used three additional corpora: Fisher [8], the ICSI Meeting corpus [9], and a 2.8-hour corpus of CB single-user sessions.

Note that training and testing on the CB multi-user corpus is always based on automatic speech recognizer output, with a word error rate of about 20%. The out-of-domain corpora,

Table 1: Sizes, distribution, and examples of in-domain utterance types: H = human-directed, C = computer-directed, M = mixed.

	Train	*Test*
Utterances	2577	2889
Recognized words	7026	7874
H utterances	40.8%	31.0%
C-noncommand utterances	31.9%	32.8%
C-command utterances	24.5%	32.0%
M utterances	3.7%	4.2%

Type	*Example*
H	Do you want to watch a movie?
C-noncommand	How is the weather today?
C-command	Scroll down, Go back.
M	Show me sandwich shops. oh, are you vegan?

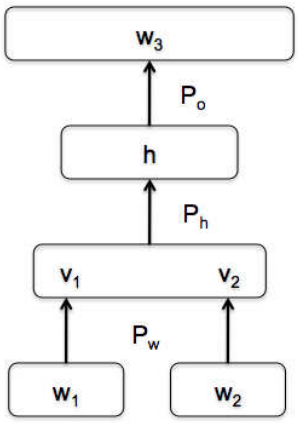

Figure 1: *The standard Neural Network Language Model. P_w, the projection layer, is shared by $\mathbf{w_1}$ and $\mathbf{w_2}$. Word embeddings from those words, v_1, v_2, are stacked into a single vector, which serves as input to a multilayer perceptron. In this architecture, the NNLM is predicting the next word, but can also predict other labels, such as addressee labels.*

however, were only available as human transcripts and used in that form.

2.2. The Neural Network Language Model

The Neural Network Language Model (NNLM), first introduced in [6], is the neural network alternative to the traditional language model. In this model, inputs are one or more words of language model history, encoded as one-hot $|V|$-dimensional vectors (i.e., one component of the vector is 1, while the rest are 0), where $|V|$ is the size of the vocabulary. Input words share a projection layer, which maps a word vector $\mathbf{w_i}$ into a word embedding $\mathbf{v_i} \in R^n$, via matrix multiplication:

$$\mathbf{v_i} = P_w \mathbf{w_i}$$

$P \in R^{n \times |V|}$, where $n \ll |V|$; n is typically around 100-1000, depending on the task. These words embeddings are concatenated into a single vector, which serve as input to a standard multi-layer perceptron (MLP). The architecture of the NNLM is shown in Figure 1. One important property of the NNLM is its ability to learn word embeddings $\mathbf{v_i}$. One can learn word embeddings via standard language modeling techniques, such as predicting the current word given the previous two, but one can also learn embeddings using addressee labels. Moreover, word embeddings from one task can improve performance on another task.

2.3. Baseline Methods

There are a number of ways to apply the NNLM for addressee detection. Perhaps the most straighforward is to adopt the approach used in standard language modeling for utterance classification: we train separate models for each class, denoted as $P_H(\mathbf{w})$ and $P_C(\mathbf{w})$, for human- or computer-addressed speech, respectively. At test time, we compute the length-normalized log-likelihood ratio

$$S = \frac{1}{N} \log \frac{P_C(\mathbf{w})}{P_H(\mathbf{w})},$$

where N is the utterance length in number of words. The score S can be compared against a threshold to decide whether an utterance is computer-directed (such a threshold would be application dependent, chosen according to the desired trade-off between false alarm and miss errors).

Our first model is obtained by simply estimating the class-specific LMs using NNLMs. One model is trained on each subset, and the score function is computed as above. We denote this approach **NNLM-ngram**. We compare this against a baseline system in which $P_H(\mathbf{w})$ and $P_C(\mathbf{w})$ are estimated by a maximum entropy-smoothed trigram model [10], the same lexical model as used in [1], and denoted here by **word-ngram**.

For the second baseline method, we note that since the neural net is fundamentally a classifier, it makes sense to train it to predict the addressee labels directly. We consider each word N-gram to be a sample to be classified. More formally, let $\mathbf{w_{i,j}}$ be the j-th N-gram for utterance i, $N_a(\mathbf{w_{i,j}}) \in R^2$ to be the addressee posterior probability of the NNLM, and a_i to be the label at utterance i. Then training involves minimizing the cross-entropy $CE = \sum_{i=1}^{N} \sum_{j=1}^{N_i} -\log[N_a(\mathbf{w_{i,j}})]_{a_i}$. At test time, the word-level posterior probabilities are averaged to obtain the utterance level detection score.

$$S = \frac{1}{N} \sum_{j=1}^{N_i} [N_a(\mathbf{w_{i,j}})]_C$$

where the subscript C refers to the computer-directed utterance class. We denote model this as **NNLM-addressee**.

Neither of the above NNLM-based methods, however, leverage an important property observed by other researchers [11, 12, 13] that word embeddings trained on one task can improve performance on a different one. In the case of the NNLM-ngram, these embeddings are only trained on human- or computer-directed speech, but not both. In the NNLM-addressee approach, word embeddings learned from a language modeling task could improve performance on the addressee one. With this in mind, we tested three different approaches, and combinations thereof.

2.4. Joint Training

One method to improve performance is to jointly predict addressee and word labels, and learn better parameters through back-propagation. Jointly predicting addressee and word labels requires us to augment our NNLM slightly. There are two ways to extend the NNLM. The first is to incorporate a second output layer, as shown in the left pane of Figure 2. For this approach, given addressee and word labels a_i and $\hat{w}_i$ respectively, and

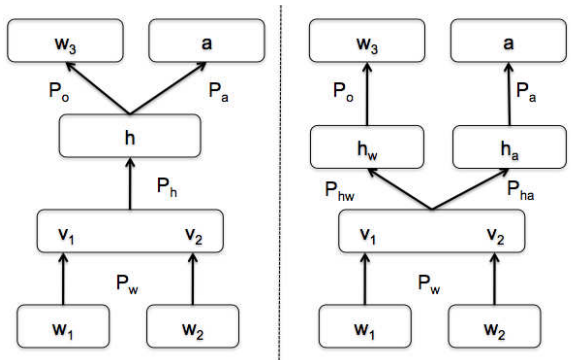

Figure 2: *Models for joint training. The left pane shows joint training with separate output parameters for word and addressee labels, while the right pane shows joint structure with separate word-vector to hidden weights.*

addressee and word NNLM posteriors vectors $\mathbf{N_a}(\mathbf{w_{i,j}})$ and $\mathbf{N_{\hat{w}}}(\mathbf{w_{i,j}})$ for n-gram $\mathbf{w_{i,j}}$, respectively, the error criterion is a weighted mix of cross-entropies:

$$CE = \sum_{i=1}^{N} \sum_{j=1}^{N_i} -\alpha \log[\mathbf{N_a}(\mathbf{w_{i,j}})]_{a_i} - (1-\alpha) \log[\mathbf{N_{\hat{w}}}(\mathbf{w_{i,j}})]_{\hat{w}_i}$$

where α is a user-defined constant (empirically, we have observed that 0.6 yields the best performance). This is denoted as **NNLM-multicriterion**.

Another method to jointly predict labels is to split the joint hidden layer into separate ones. This leads to a second structure, in which both the output layer and the word embedding to hidden layer parameters are separate, as shown in the right pane of Figure 2. The error criterion is identical to the NNLM-multicriterion is denoted as **NNLM-shared_projection**.

2.5. Two-Phase and Out-of-Domain Training

Another way to learn potentially better models is to train on different criteria serially. In this approach, we decouple the training into two phases, one for learning word embeddings, and a one for learning utterance classificiation. In Phase 1, we train the NNLM to predict word labels; in Phase 2, using Phase 1 weights as initialization and replacing the output layer to predict addressee labels, we update the NNLM parameters on the error signal from the addressee labels. This is denoted as **NNLM-two_phase-addressee**. Moreover, we can perform joint optimization as described above, denoting **NNLM-two_phase-multicriterion** and **NNLM-two_phase-shared_projection** for models referenced in the left and right panes of Figure 2, respectively.

Training on word prediction and addressee labels serially also allows us to train on out-of-domain corpora. Given that better word embeddings can improve performance across tasks, a natural question to ask is whether Phase 1 training on a larger corpus can improve results on the addressee task. To investigate this question, we included ASR output from three corpora in addition to the CB multi-user data: Fisher, ICSI meeting corpus, and the CB sessions recorded with a single user. For one set of experiments, we included all corpora for Phase 1 training, while for a second set, we used only Fisher.

For out-of-domain training, one must take particular care in selecting the vocabulary, as the vocabulary size changes between corpora. For Phase 1 training that includes all corpora, we include the 8,000, 5,000, and 2,000 most frequent words from the Fisher, ICSI meeting corpus, and CB single-user respectively. Any word that was not found within the vocabulary is mapped to a special "out of vocabulary" token. Frequent words from CB multi-user were not included in the list so that the "out of vocabulary" token occurred in Phase 2 training. The vocabulary comprises 10,808 words, including start and stop tags, and the "out of vocabulary" token. Systems using all out-of-domain corpora in Phase 1 are denoted with **two_phase-A** instead of **two_phase**.

For Fisher-only Phase 1 training, applying this same vocabulary would result in words which only occurred in Phase 2 training, or occurred only in test. In the first case, the word embeddings would be suboptimal, while in the latter case, word embedding would be completely random as they are unlearned. For Phase 1 training with the Fisher corpus, we mapped unseen and once-seen words in Phase 1 training to the "out of vocabulary" token. The vocabulary size decreased slightly to 10,500 words. Systems using only Fisher in Phase 1 are denoted with **two_phase-F**.

2.6. Experimental Setup

All neural network language models use as input a bigram, encoded as two one-hot $|V|$-dimensional vectors. The projection layer embeds each word vector into a 500-dimensional space, as empirically, this dimensionality seems to give the best results. For the standard NNLM and the NNLM-multicriterion models, the number of hidden units is 1000, while for NNLM-shared_projection, there are two 1000 hidden unit layers. The NNLMs are trained with stochastic gradient descent, with batch size 256. For single-phase NNLMs, training ran for 90 epochs, with 30 each at learning rates 0.008, 0.004, and 0.002.

For NNLMs trained with two phases, Phase 2 training uses the identical optimization as the single-phase NNLMs. For Phase 1, the batch size was 256 for CB-Multi and 16384 for out-of-domain corpora, and the NNLM is trained for 5 epochs at learning rate 0.08. Finally, the baseline NNLM-gram systems share the same architecture as the standard NNLM, and is trained for 40 epochs, with 20 each at learning rates 0.008 and 0.004.

For model combination and evaluation, we use linear logistic regression (LLR) to calibrate all model scores or to combine multiple scores where applicable [14]. To estimate LLR parameters, we jack-knife over all sessions in the test data, training on all but one session in turn, and cycling through all sessions. Scores are then pooled over the entire test set and evaluated using equal error rate (EER). The EER is obtained by choosing a decision threshold that equates false alarm and miss error probabilities. EER is thus independent of class priors. The behavior of the various systems is also more fully characterized by a detection error tradeoff (DET) curve, showing how a moving decision threshold changes the two error type rates.

3. Results and Discussion

3.1. Results

Results for the different Neural Network Language Models is shown in Table 2. Neither of the NNLM baseline approaches performs as well as the standard word-ngram. NNLMs trained on addressee labels is over 3% worse, while the NNLM-ngram is 2.46% worse. Jointly training on word and addressee labels improves performance over the Neural Network Language Model baseline approaches, with the NNLM-shared_projection system about .62% better than the multicriterion one. Neither

of the joint training approaches, however, beats the maximum entropy language model baseline.

Two-phase training improves upon the NNLM-addressee baseline by 3.25%, and improves upon the word-ngram baseline by 0.24%, with an EER of 26.76%.[1] Including the Fisher corpus and all corpora in Phase 1 improves EER by 0.35% and 0.22%, respectively. When performing joint training in Phase 2, multicriterion models far outperform shared projection systems, even though shared projection systems are better for single-phase training. In fact, shared projection systems are worse than simple addressee detection in two-phase systems.

We surmise that Phase 1 already optimizes word embeddings for the word prediction task, and any extra training that effectively learns poorer word embeddings on a smaller corpus (we don't necessarily care about the other parameters of the word prediction part of the shared projection system, since only the addressee prediction is used after Phase 2) in fact hurts performance. Multicriteria training, which jointly updates both the word embedding and the parameters from the word embedding to the hidden layer, improve upon the addressee baseline, and give us the best single system at 26.23% EER, which is 0.77% better than the baseline.

The real contribution of the NNLM systems to addressee detection comes when combining the two approaches. Combining the standard maximum entropy language model with a single two-phase system improves upon the LM baseline by over 2% in the best case (two-phase system trained on Fisher in Phase 1 and multicriterion training in Phase 2). Including the two-phase multicriterion system trained on Fisher, ICSI meeting corpus, and CB single user gave us our best result at 24.75%. Figure 3 shows the DET plot for baseline and better performing systems.

System	EER
Baseline Systems	
1. word-ngram	27.00%
2. NNLM-ngram	29.46%
3. NNLM-addressee	30.01%
Joint Systems	
4. NNLM-multicriterion	28.98%
5. NNLM-shared_projection	28.32%
Two-Phase Systems	
6. NNLM-two_phase-addressee	26.76%
7. NNLM-two_phase-F-addressee	26.41%
8. NNLM-two_phase-F-multicriterion	26.33%
9. NNLM-two_phase-F-shared_projection	26.65%
10. NNLM-two_phase-A-addressee	26.54%
11. NNLM-two_phase-A-multicriterion	**26.23%**
12. NNLM-two_phase-A-shared_projection	26.76%
Combination Systems	
13. System 1. + System 10.	25.64%
14. System 1. + System 11.	25.41%
15. System 1. + System 7.	25.22%
16. System 1. + System 8.	**24.96%**
17. System 1. + System 8. + System 11.	**24.75%**

Table 2: *Equal Error Rate Performance for Neural Network Language Modeling systems for addressee detection. The EERs of notable systems are highlighted.*

[1] All two-phase systems are trained for 5 epochs in Phase 1. As shown in Figure 4, 5 epochs seems to produce the best results on addressee detection.

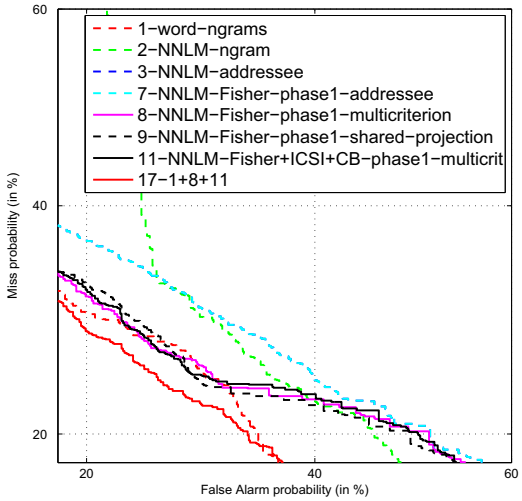

Figure 3: *Detection Error Tradeoff for baseline and better performing systems.*

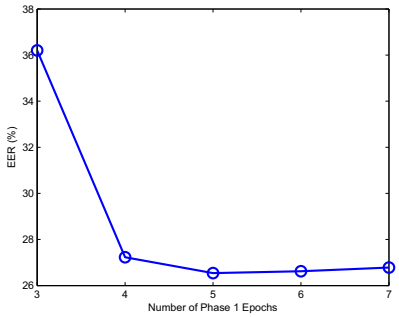

Figure 4: *Effect of Phase 1 Training on Addressee Detection Task. The x-axis denotes the number of Phase 1 training epochs on Fisher, ICSI, and CB corpora, and y-axis is EER.*

4. Conclusions

We have investigated whether neural networks can enhance the lexical classification of addressees in a human-human-computer dialog system, replacing the standard N-gram language models. We found no improvement from a NNLM replacement of the N-gram LM as class likelihood estimators (possibly due to the small training corpus), but found gains from a modified NNLM that learns distributed word representations in a first training phase, and is trained on the utterance classification task in a second phase. It is also advantageous to combine word prediction and classification during the second phase. The single best NNLM system reduced equal error rate only modestly (0.7%) from the baseline (27.0%), but we obtained a substantial error reduction (2.0%) by combining the N-gram model with three NNLMs that employed diverse initialization data and training criteria.

In future work we plan to explore other neural language modeling architectures, in particular recurrent neural nets [15], which have been shown superior to regular NNLMs in the language modeling task.

5. References

[1] E. Shriberg, A. Stolcke, and S. Ravuri, "Addressee detection for dialog systems using temporal and spectral dimensions of speaking style", *in Proc. Interspeech*, Lyon, Aug. 2013.

[2] M. Katzenmaier, R. Stiefelhagen, and T. Schultz, "Identifying the addressee in human-human-robot interactions based on head pose and speech", *in Proceedings of the 6th international conference on Multimodal interfaces*, pp. 144–151, State College, PA, USA, 2004. ACM.

[3] R. op den Akker and D. Traum, "A comparison of addressee detection methods for multiparty conversations", *in Proceedings of Diaholmia*, pp. 99–106, 2009.

[4] D. Bohus and E. Horvitz, "Multiparty turn taking in situated dialog: Study, lessons, and directions", *in Proceedings ACL SIGDIAL*, pp. 98–109, Portland, OR, June 2011.

[5] N. Baba, H.-H. Huang, and Y. I. Nakano, "Addressee identification for human-human-agent multiparty conversations in different proxemics", *in Proceedings 4th Workshop on Eye Gaze in Intelligent Human Machine Interaction*. ACM, Oct. 2012, Article no. 6.

[6] Y. Bengio, H. Schwenk, J.-S. Senécal, F. Morin, and J.-L. Gauvain, "Neural probabilistic language models", *in Studies in Fuzziness and Soft Computing*, vol. 194, pp. 137–186. 2006.

[7] L. Heck, D. Hakkani-Tür, M. Chinthakunta, G. Tur, R. Iyer, P. Parthasarathy, L. Stifelman, A. Fidler, and E. Shriberg, "Multimodal conversational search and browse", *in Proceedings IEEE Workshop on Speech, Language and Audio in Multimedia*, Marseille, Aug. 2013.

[8] C. Cieri, D. Miller, and K. Walker, "The Fisher corpus: a resource for the next generations of speech-to-text", *in Proceedings 4th International Conference on Language Resources and Evaluation*, pp. 69–71, Lisbon, May 2004.

[9] A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters, "The ICSI Meeting Corpus", *in Proc. ICASSP*, vol. 1, pp. 364–367, Hong Kong, Apr. 2003.

[10] T. Alumäe and M. Kurimo, "Efficient estimation of maximum entropy language models with N-gram features: An SRILM extension", *in Proc. Interspeech*, pp. 1820–1823, Portland, Oregon, Sep. 2012.

[11] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning", *in Proc. 25th International Conference on Machine Learning*, 2008.

[12] J. Turian, D. D. Et, R. O. (diro, U. D. Montral, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semisupervised learning", *in In ACL*, pp. 384–394, 2010.

[13] M. thang Luong, R. Socher, and C. D. Manning, "Better word representations with recursive neural networks for morphology", *in Conference on Computational Natural Language Learning coNLL '13*, 2013.

[14] S. Pigeon, P. Druyts, and P. Verlinde, "Applying logistic regression to the fusion of the NIST'99 1-speaker submissions", *Digital Signal Processing*, vol. 10, pp. 237–248, Jan. 2000.

[15] T. Mikolov, M. Karafiát, L. Burget, J. H. Černocký, and S. Khudanpur, "Recurrent neural network based language model", *in Proc. Interspeech*, pp. 1045–1048, Makuhari, Japan, Sep. 2010.