Recurrent Neural Networks for Language Understanding

Kaisheng Yao[1], Geoffrey Zweig[2], Mei-Yuh Hwang[1], Yangyang Shi[3], and Dong Yu[2]

[1]Online Services Division, Microsoft, China
[2]Microsoft Research, USA
[3]Interactive Intelligence Group, Delft University of Technology, NL
{kaisheny, gzweig, mehwang, dongyu}@microsoft.com

Abstract

Recurrent Neural Network Language Models (RNN-LMs) have recently shown exceptional performance across a variety of applications. In this paper, we modify the architecture to perform Language Understanding, and advance the state-of-the-art for the widely used ATIS dataset. The core of our approach is to take words as input as in a standard RNN-LM, and then to predict slot labels rather than words on the output side. We present several variations that differ in the amount of word context that is used on the input side, and in the use of non-lexical features. Remarkably, our simplest model produces state-of-the-art results, and we advance state-of-the-art through the use of bag-of-words, word embedding, named-entity, syntactic, and word-class features. Analysis indicates that the superior performance is attributable to the task-specific word representations learned by the RNN.

Index Terms: Recurrent Neural Networks, Spoken Language Understanding

1. Introduction

In recent years, Recurrent Neural Network language models (RNN-LMs) have demonstrated outstanding performance in a variety of natural language processing tasks [1, 2, 3, 4, 5, 6]. In common with other continuous space language models such as feed-forward neural network LMs [7, 8, 9, 10, 11] and the Hierarchical Log-Bilinear model [12], the RNN-LM represents each word as a high-dimensional real-valued vector. Critically, in this vector space, similar words tend to be close together, and relationships between words are preserved [13]; thus, adjusting the model parameters to increase the likelihood of one word in a particular context increases the likelihood of similar words in similar contexts. In conjunction with training on large datasets, continuous space language models have resulted in improvements in language model perplexity [3, 9, 8] machine translation Bilingual Evaluation Understudy (BLEU) score [14, 15, 16] and speech recognition Word Error Rate (WER) [7, 10, 6].

In this paper, we apply recurrent neural networks to Language Understanding (LU). In classical LU systems [17, 18, 19, 20, 21, 22, 23, 24, 25], one of the key tasks is to label words with semantic meaning. For example, in the sentence "I want to fly from Seattle to Paris," the word "Seattle" should be labeled as the departure-city of a trip, and "Paris" as the arrival-city. Perhaps the most obvious approach to this task is the use of Conditional Random Fields (CRFs) [26], in which an exponential model is used to compute the probability of a label sequence given the input word sequence. The CRF produces the single, globally most likely labeling, and has been widely used in LU [21, 27, 28]. Other sequence labeling methods that have been investigated include Support Vector Machines [29], Finite State Transducers [21] and Machine Translation models [30].

To adapt the RNN-LM to the LU task, we use the classic Elman RNN architecture [31] adopted by Mikolov [1]. This architecture consists of inputs to a set of hidden nodes; a fully connected set of recurrent connections amongst the hidden nodes; and a set of output nodes. For language modeling, the network has the property that the output is simply the input word sequence shifted in time so that the network predicts the next word in the sequence; both inputs and outputs are words. To perform LU, we train the network using the semantic labels rather than the words themselves as targets. In the most basic configuration, our network has no explicit knowledge of "future words" when predicting the label of the current word, and we present an extension which does. As further extensions, we use the named entity and syntactic features present in the ATIS dataset, word embeddings trained on the ATIS dataset, as well as word-class information inferred from Wikipedia data using the Brown word classing algorithm [32]. A number of alternative architectures are discussed elsewhere in this proceedings [33].

The remainder of this paper is organized as follows. Section 2 describes the model we use, and its extensions. Section 3 presents experimental results, and Section 4 presents an analysis of what the system learned. Section 5 offers concluding remarks.

2. Recurrent Neural Networks for LU

2.1. Elman Architecture

The RNN architecture is illustrated in Figure 1, where it is "unrolled" across time to cover three consecutive word inputs. This architecture consists of an input layer at the bottom, a hidden layer in the middle with recurrent connections shown as dashed lines, and an output layer at top. Each layer represents a set of neurons, and the layers are connected with weights denoted by the matrices $\mathbf{U}$, $\mathbf{W}$, and $\mathbf{V}$. The input layer $\mathbf{w}(t)$ represents input word at time t encoded using 1-of-N coding, and the output layer $\mathbf{y}(t)$ produces a probability distribution over semantic labels. The hidden layer $\mathbf{s}(t)$ maintains a representation of the sentence history. The input vector $\mathbf{w}(t)$ has a dimensionality equal to the vocabulary size, and the output vector $\mathbf{y}(t)$ has a dimensionality equal to the number of possible semantic labels. The values in the hidden and output layers are computed as follows:

$$\mathbf{s}(t) = f\left(\mathbf{U}\mathbf{w}(t) + \mathbf{W}\mathbf{s}(t-1)\right) \quad (1)$$

$$\mathbf{y}(t) = g\left(\mathbf{V}\mathbf{s}(t)\right), \quad (2)$$

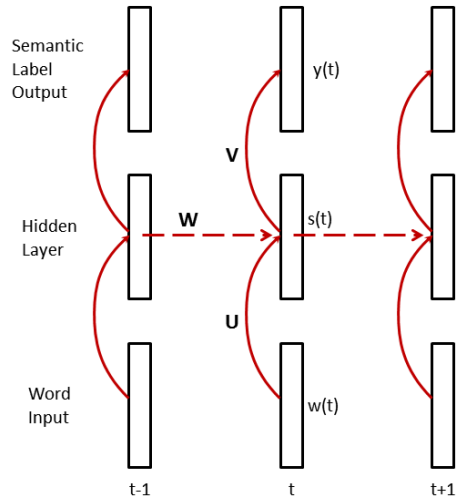

Figure 1: Recurrent Neural Network Model for Language Understanding.

where

$$f(z) = \frac{1}{1+e^{-z}}, \quad g(z_m) = \frac{e^{z_m}}{\sum_k e^{z_k}}. \tag{3}$$

The model is trained using standard back-propagation to maximize the data conditional likelihood:

$$\prod_t P(\mathbf{y}(t)|\mathbf{w}(1)\dots\mathbf{w}(t)) \tag{4}$$

Note that this model has no direct interdependence between slot output values; the probability distribution is strictly a function of the hidden layer activations, which in turn depend only on the word inputs (and their own past values). Thus, the likeliest sequence of semantic labels can be output with a series of online decisions:

$$\mathbf{y}^\star(t) = \arg\max P(\mathbf{y}(t)|\mathbf{w}(1)\dots\mathbf{w}(t)) \tag{5}$$

This has the advantage of being online and very simple; it is unnecessary to do a dynamic programming search over labeling to find the optimum.

2.2. Incorporating Future Word Observations

Since our task is to find the likeliest label sequence given all the words in the input, we may legitimately use "future" words as input when determining the semantic label for word $\mathbf{w}(t)$. We have considered two methods for doing this. In the first, we change the input layer from a "one-hot" representation to an "n-hot" or bag-of-words (BoW) representation in which there is a non-zero value for not just the current word, but the next $n-1$ words as well. This has the advantage of using greater context, but the disadvantage that ordering information is lost. To address this, we have also used the "feature-augmented" architecture of Figure 2, as proposed in [5, 34]. In this approach, side-information is provided to the network via an extra layer of dense (as opposed to 1-hot) inputs $\mathbf{f}(t)$ with connection weights $\mathbf{F}$ to the hidden layer and $\mathbf{G}$ to the output layer. In our case, we provide continuous space vector representations of the future words as input. The representation of a word is learned by a non-augmented network (it consists of the weights from the input to the hidden layer) as described in [1]. To keep word ordering information, these representations are concatenated in sequence in a given context window. The training and decoding procedures are otherwise unaltered.

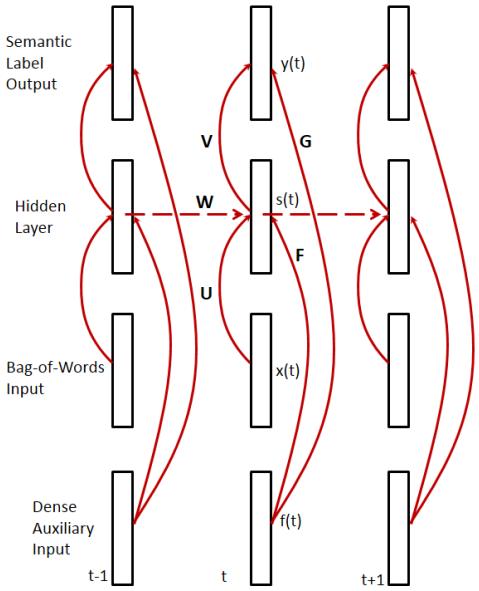

Figure 2: Adding information to the basic model.

In this model, we modify the activation computation as follows:

$$\mathbf{s}(t) = f\left(\mathbf{U}\mathbf{x}(t) + \mathbf{W}\mathbf{s}(t-1) + \mathbf{F}\mathbf{f}(t)\right) \tag{6}$$

$$\mathbf{y}(t) = g\left(\mathbf{V}\mathbf{s}(t) + \mathbf{G}\mathbf{f}(t)\right), \tag{7}$$

where $\mathbf{x}(t)$ now can be either $\mathbf{w}(t)$ or a bag-of-words vector. For instance, $\mathbf{x}(t) = \{\mathbf{w}(t), \mathbf{w}(t+1)\}$ and it consists of the current and the next words, forming a "2-hot" representation.

2.3. Incorporating A-Priori Word Information

The ATIS dataset provides two forms of input besides the words themselves: named-entity tags for the words, and syntactic labels. Previous researchers [35] have found it useful to exploit this information, and it is easily incorporated in the RNN approach. We have done this with the architecture of Figure 2 by making $\mathbf{f}(t)$ to be a concatenation of "one-hot" input vectors. Each input vector represents the named-entity and/or syntactic label of a word in a context window. This is feasible since the number of such labels is small; thus the number of dimensions in the dense $\mathbf{f}$ vector is manageable.

As an additional experiment, we have considered the addition of information to the system via classes derived from the Wikipedia dataset. To do this, we sub-sampled 70M words from Wikipedia, and used the clustering criterion of Brown et al. [32] to group the words into 200 classes. These were then used rather than the named entity labels. We have also used $\mathbf{f}(t)$ to include both the word embedding and "one-hot" side information. We found that the continuous valued word embeddings need to be normalized to unit-length to avoid numerical issues.

3. Experiments

3.1. ATIS Dataset

In order to be able to compare with previously studied methods, we report results with the widely used ATIS dataset [17, 36]. This dataset focuses on the air travel domain, and consists of

I	want	to	fly	to	Boston	tomorrow
-	-	-	-	-	Dest	ArDay

Table 1: Examples of labeled sentences. Label names have been shortened to fit. Many words are labeled "null" or "-".

audio recordings of people making travel reservations, and semantic interpretations of the sentences. In this database, the words in each sentence are labeled with their value with respect to certain semantic frames. Table 1 shows an example of an annotated sentence.

The training data consists of 4978 sentences and 56590 words. Test data consists of 893 sentences and 9198 words. The number of distinct slot labels is 127, including the common "null" label; there are a total of 25509 non-null slot occurrences in the training and testing data respectively. In addition to the words themselves, the ATIS set provides named-entity and syntactic labels. There are 132 named entity labels [1], e.g. "B-city_name," "B-time" and "I-time," and 38 standard syntactic labels, e.g. "NNP," "VBP," and "JJ." To deal with unseen words in test set, we marked any words with only one occurrence in the training data as <unk> and use this label to represent those unseen words in the validation and test sets. Based on the number of words in the dataset and assuming independent errors, changes of approximately 0.6% in F1 measure are significant at the 95% level.

3.2. Results with Lexical Features

Figure 3 shows results using lexical features, using just the current word as the input when predicting a slot value, and when using the next two words as well. Initial experiments showed no significant gain from looking more than two words ahead with ordered lookahead; thus the plots are for two-word lookahead. Unordered lookahead was done by using a 3-hot input representation, while ordered lookahead was done with the side-information channel of Section 2.2. To obtain the continuous space embedding of words for use in the side-channel, we trained a standard RNN-LU model with a hidden layer size of 200, and used the input layer weights as the word representations [13]. For two words of lookahead, we concatenated the word vectors, forming a 400 dimensional side-channel. We see that using two words of lookahead is significantly better than not, especially for small hidden layer sizes. Hidden layer sizes of 100 and more all perform similarly. Also shown in Figure 3 is the result of adding the class label of the current word, as learned from the Wikipedia corpus. This was added as an additional 1-hot input.

3.3. Adding Non-Lexical Features

Figure 4 shows results using the named-entity and syntactic tags included in the database. The named entity features provide a large boost, typically 3% in F1 measure, and double that when a small network is used. These features are highly related to the final output labels, and hand-determined, so this is not surprising. The syntactic part-of-speech tags are not directly related to the output labels, and provide relatively little benefit.

3.4. Bag-of-Words Effects

One originality of this work is to use a bag-of-words input to the RNN. To verify effectiveness of using a bag-of-words, we

[1]These include the "null" label for words other than named-entity and the beginning and ending sentence labels.

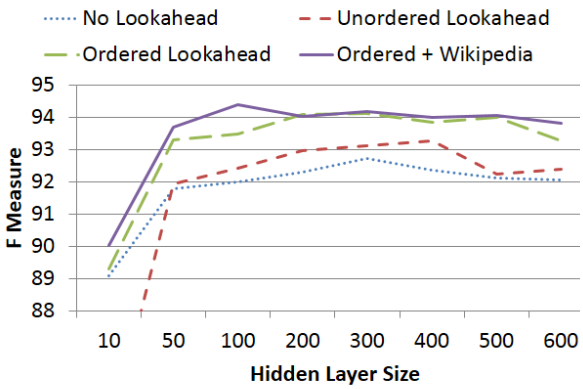

Figure 3: F measure for different hidden layer sizes, using lexical features only.

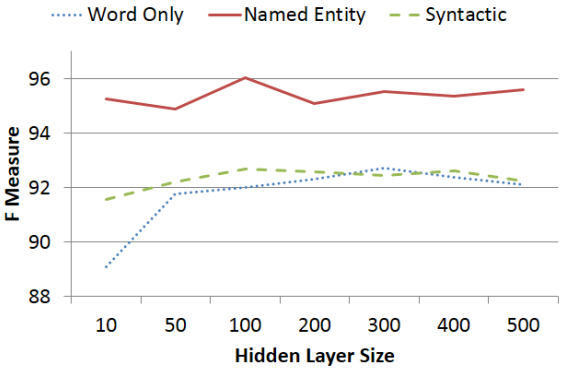

Figure 4: F measure for different hidden layer sizes, using non-lexical features. All conditions use one word lookahead for the lexical feature only.

conducted experiments on top of using ordered lookahead and Wikipedia class information. Table 2 shows the F1 score of using an ordered lookahead of two future words, each represented as a 200 dimension word embedding learned from the standard RNN-LU model mentioned in Sec. 3.2. Wikipedia class is also the same as in Sec. 3.2. Results in the table show that F1 score can be further improved using bag-of-words input, compared against using the current word as input. For example, when hidden layer dimension is 200, the F1 score is increased to 94.79% with the addition of "2-hot" input, from 94.01% with the current word as input.

3.5. Comparison with Past Results

ATIS is a well-studied dataset, and Table 3 presents our results alongside benchmark numbers. The result of [37] is the best previously published result with lexical plus Wikipedia [38] features, with a slot filling F1 score of 91.88% using a kernel deep convex network; our comparable result is 93.61% if using the embedding from [38] and 94.39% if using Wikipedia class information described in Sec. 2.3. Using bag-of-words input, our result is further improved to 94.91%. The result of [35], which is 95.0% F1 score, is the best previously published result with lexical plus named-entity (NE) feature; ours is 96.04% with word plus NE feature in a context window of length 3. As

BoW $\mathbf{x}(t)$	100	200	300	400
$\{\mathbf{w}(t)\}$	94.39	94.01	94.18	93.99
$\{\mathbf{w}(t), \mathbf{w}(t+1)\}$	93.71	94.79	94.45	94.55
$\{\mathbf{w}(t), \mathbf{w}(t+1), \mathbf{w}(t+2)\}$	94.26	94.91	94.11	94.27

Table 2: Effects of using Bag-of-Words input in addition to ordered lookahead, as a function of hidden layer size.

Method	Features	F1 (in %)
FST [21]	Lex	91.87
SVM [21]	Lex	89.76
CRF [37]	Lex	91.09
RNN [This work]	Lex	**94.11** [2]
KDCN [37]	Lex + Wikipedia	91.88
RNN [This work]	Lex + Wikipedia	**94.91** [3]
CRF [35]	Lex + NE	94.4
CRF [35]	Lex + NE+Syntactic+SentSimp	95.0
SVM [21]	Lex + Prior Knowledge	95.74
RNN [This work]	Lex + NE	**96.60** [4]

Table 3: Comparison with Previous Results.

is evident in the table, the RNN approach advances the state-of-the-art over previous techniques. The highest score we obtained was 96.60%, which used bag-of-words and named-entity features.

4. What the Model Learned

In contrast to previous CRF approaches, the RNN approach implicitly learns a task-appropriate continuous space representation for each word in the ATIS set, and we hypothesize that learning this task-specific representation accounts for the method's improved performance. This representation is present in the weights connecting the one-hot input layer to the hidden layer, and learned through back propagation to maximize the likelihood of the semantic output labels. To understand it better, we use t-SNE [39] to plot frequent words that are in the ten most common slots in two-dimensional space, as shown in color in Figure 5. Perhaps the most striking feature of this visualization is that in the projection plane, the words that are labeled as background (null) in the dataset have been completely separated from the others, and restricted to the bottom-left half-plane. In the upper-right half plane, the other words are clearly clustered by semantic label.

We have made two quantitative studies of this phenomenon. First, we computed the Fisher discriminant value of two projections: that learned by training the RNN to do LU, and that learned by training the RNN as a language model. This is computed as the ratio of the between-class variance to the within-class variance. When trained to do LU, the values for the first and second dimensions are 10.5 and 32.6 respectively. In contrast, when we use the representations learned in a generic language model trained on the ATIS data, the ratios are only 7.6 and 7.0 respectively. This indicates that the task specific training indeed focuses the representations to distinguish relevant concepts.

[2] with word embedding learned from ATIS

[3] with word embedding learned frm ATIS, Wikipedia class, and bag-of-words input

[4] with bag-of-words $\{\mathbf{w}(t), \mathbf{w}(t+1), \mathbf{w}(t+2)\}$ and NE feature in a context window of length 3

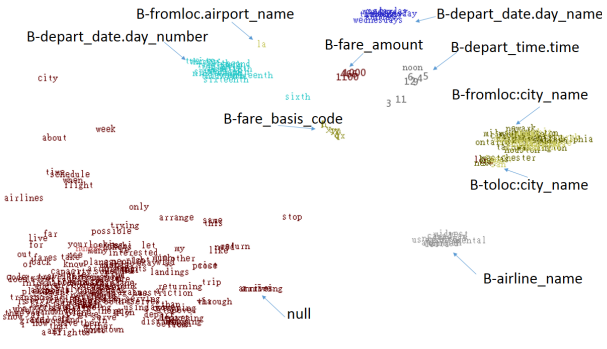

Figure 5: Learned word representations. Note the separation between labeled and unlabeled (null) words.

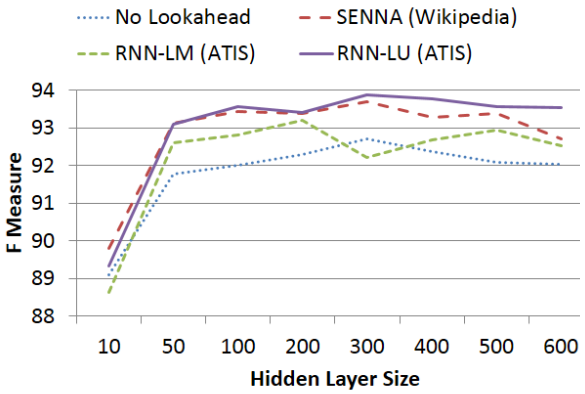

Figure 6: Improvement from task-specific word embeddings.

As a second experiment, we trained RNN-LU systems using three different embeddings to represent the future words as in Section 2.2: a task-specific embedding learned from training an RNN-LU system on the ATIS data; a language-model (RNN-LM) embedding also trained with the ATIS data; and finally the generic SENNA embeddings derived from the Wikipedia corpus [38]. Performance with the SENNA features was not quite as good as with the discrete classes of Section 2.3, but they provide a Wikipedia-based embedding for comparison. For consistency with SENNA, 50 dimensional embeddings were used in all cases. As can be seen in Figure 6, the RNN-LU embeddings are the most effective for the LU task. After that, the SENNA embeddings, trained on the much larger Wikipedia corpus are the most effective, followed by the RNN-LM embeddings. Thus, both qualitative and quantitative evaluations show that the task-specific word representations learned by the RNN are effective and behave differently from other representations. The similar argument and observation has been made in deep neural networks applied to speech recognition tasks[40, 41].

5. Conclusion

We have presented an adaptation of Recurrent Neural Networks to perform a language understanding task. In contrast to CRFs, the basic RNN approach we evaluate makes a sequence of local labeling assignments. Remarkably, it outperforms previous CRF results by a large margin, as well as previous neural-net based approaches. We show that the model learns task appropriate word-embedding, and hypothesize that this accounts for the improved performance.

6. References

[1] T. Mikolov, M. Karafiat, J. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH*, 2010, pp. 1045–1048.

[2] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network based language model," in *ICASSP*, 2011, pp. 5528–5531.

[3] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocky, "Empirical evaluation and combination of advanced language modeling techniques," in *INTERSPEECH*, 2011, pp. 605–608.

[4] T. Mikolov, A. Deoras, D. Povey, L. Burget, and J. Cernocky, "Strategies for training large scale neural network language models," in *ASRU*, 2011.

[5] T. Mikolov and G. Zweig, "Context dependent recurrent neural network language model," in *Proc. SLT*, 2012, pp. 234–239.

[6] E. Arisoy, T. N. Sainath, B. Kingsbury, and B. Ramabhadran, "Deep neural network language models," in *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, 2012, pp. 20–28.

[7] H. Schwenk and J.L. Gauvain, "Connectionist language modeling for large vocabulary continuous speech recognition," in *ICASSP*. IEEE, 2002, vol. 1, pp. I–765.

[8] Y. Bengio, R. Ducharme, Vincent, P., and C. Jauvin, "A neural probabilistic language model," *Journal of Machine Learning Reseach*, vol. 3, no. 6, 2003.

[9] H. Schwenk, "Continuous space language models," *Computer Speech and Language*, vol. 21, no. 3, pp. 492 – 518, 2007.

[10] H.-S. Le, I. Oparin, A. Allauzen, J.-L. Gauvain, and F. Yvon, "Structured output layer neural network language model," in *ICASSP*, 2011, pp. 5524–5527.

[11] F. Morin and Y. Bengio, "Hierarchical probabilistic neural network language model," in *Proceedings of the international workshop on artificial intelligence and statistics*, 2005, pp. 246–252.

[12] A. Mnih and G.E. Hinton, "A scalable hierarchical distributed language model," in *Advances in neural information processing systems*, 2009, vol. 21, pp. 1081–1088.

[13] T. Mikolov, W.T. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *NAACL-HLT, to appear*, 2013.

[14] H. Schwenk, D. Dchelotte, and J.-L. Gauvain, "Continuous space language models for statistical machine translation," in *COLING/ACL*, 2006.

[15] L. H. Son, A. Allauzen, and F. Yvon, "Measuring the influence of long range dependencies with neural network language models," in *NAACL/HLT*, 2012, pp. 1–10.

[16] H. Schwenk, A. Rousseau, and M. Attik, "Large, pruned or continuous space language models on a GPU for statistical machine translation," in *NAACL/HLT*, 2012, pp. 723–730.

[17] C. Hemphill, J. Godfrey, and G. Doddington, "The ATIS spoken language systems pilot corpus," in *Proceedings of the DARPA speech and natural language workshop*, 1990, pp. 96–101.

[18] P. Price, "Evaluation of spoken language systems: The ATIS domain," in *Proceedings of the Third DARPA Speech and Natural Language Workshop*. Morgan Kaufmann, 1990, pp. 91–95.

[19] W. Ward et al., "The CMU air travel information service: Understanding spontaneous speech," in *Proceedings of the DARPA Speech and Natural Language Workshop*, 1990, pp. 127–129.

[20] Y. He and S. Young, "A data-driven spoken language understanding system," in *ASRU*, 2003, pp. 583–588.

[21] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," *INTERSPEECH*, pp. 1605–1608, 2007.

[22] R. De Mori, "Spoken language understanding: A survey," in *ASRU*, 2007, pp. 365–376.

[23] F. Béchet, "Processing spontaneous speech in deployed spoken language understanding systems: a survey," *SLT*, vol. 1, 2008.

[24] V. Zue and J. Glass, "Conversational interfaces: advances and challenges," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1166–1180, 2000.

[25] J. Liu, S. Cyphers, P. Pasupat, I. McGraw, and J. Glass, "A conversational movie search system based on conditional random fields," in *INTERSPEECH*, 2012.

[26] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," in *ICML*, 2001, pp. 282–289.

[27] Y.-Y. Wang, A. Acero, M. Mahajan, and J. Lee, "Combining statistical and knowledge-based spoken language understanding in conditional models," in *COLING/ACL*, 2006, pp. 882–889.

[28] A. Moschitti, G. Riccardi, and C. Raymond, "Spoken language understanding with kernels for syntactic/semantic structures," in *ASRU*, 2007, pp. 183–188.

[29] T. Kudo and Y. Matsumoto, "Chunking with support vector machines," in *ACL*, 2001.

[30] K. Macherey, F. Och, H. Ney, et al., "Natural language understanding using statistical machine translation," in *European Conf. on Speech Communication and Technology*. Citeseer, 2001, pp. 2205–2208.

[31] J. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[32] P.F. Brown, V.J. Della Pietra, P.V. deSouza, J. Lai, and R.L. Mercer, "Class-based n-gram models of natural language," *Computational Linguistics*, vol. 18, no. 4, pp. 467–479, 1992.

[33] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for language understanding," in *submitted to INTERSPEECH*, 2013.

[34] Y. Shi, P. Wiggers, and C. M. Jonker, "Towards recurrent neural networks language models with linguistic and contextual features," in *INTERSPEECH*, 2012.

[35] G. Tur, D. Hakkani-Tur, L. Heck, and S. Parthasarathy, "Sentence simplification for spoken language understanding," in *ICASSP*, 2011, pp. 5628–5631.

[36] D. Dahl, M. Bates, M. Brown, W. Fisher, K. Hunicke-Smith, D. Pallett, C. Pao, A. Rudnicky, and E. Shriberg, "Expanding the scope of the ATIS task: The ATIS-3 corpus," in *Proceedings of the workshop on Human Language Technology*. Association for Computational Linguistics, 1994, pp. 43–48.

[37] L. Deng, G. Tur, X. He, and D. Hakkani-Tur, "Use of kernel deep convex networks and end-to-end learning for spoken language understanding," in *SLT*, 2013, pp. 210–215.

[38] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[39] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, 2008.

[40] F. Seide, G. Li, X. Chen, and D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *ASRU*, 2011, pp. 24–29.

[41] D. Yu, M. L. Seltzer, J. Li, J.-T. Huang, and F. Seide, "Feature learning in deep neural networks - studies on speech recognition tasks," 2013, http://arxiv.org/pdf/1301.3605.