Efficient learning for spoken language understanding tasks with word embedding based pre-training

Yi Luan , Shinji Watanabe, and Bret Harsham

Mitsubishi Electric Research Laboratories

luanyi@u.washington.edu, {watanabe,harsham}@merl.com

Abstract

Spoken language understanding (SLU) tasks such as goal estimation and intention identification from user's commands are essential components in spoken dialog systems. In recent years, neural network approaches have shown great success in various SLU tasks. However, one major difficulty of SLU is that the annotation of collected data can be expensive. Often this results in insufficient data being available for a task. The performance of a neural network trained in low resource conditions is usually inferior because of over-training. To improve the performance, this paper investigates the use of unsupervised training methods with large-scale corpora based on word embedding and latent topic models to pre-train the SLU networks. In order to capture long-term characteristics over the entire dialog, we propose a novel Recurrent Neural Network (RNN) architecture. The proposed RNN uses two sub-networks to model the different time scales represented by word and turn sequences. The combination of pre-training and RNN gives us a 18% relative error reduction compared to a baseline system.

Index Terms: spoken language understanding, fine-tuning, semantic embedding, recurrent neural networks, goal estimation

1. Introduction

Spoken language understanding (SLU) in dialog systems extracts semantic information from the output of an automatic speech recognizer (ASR) [1, 2]. The dialog manager (DM) then determines the next machine action given the SLU output. In the last decade, a variety of practical goal-oriented spoken dialog systems have been built for different tasks [3, 4, 5, 6]. This paper focuses on two key tasks in targeted dialog and understanding applications: user intention understanding and user goal estimation.

User intention understanding is the extraction of the intended meaning (called "intention" hereafter) of one user utterance, performed by the SLU module. The dialog manager determines which action to take next based on the result of intention understanding.

User goal estimation is a similar concept, but is an estimation of the final system action (called "goal" hereafter) that the user wishes to accomplish during the dialog. A dialog usually consists of a series of user utterances and system actions, so the goal estimation takes place over a longer time scale than user intention understanding. The system's estimate of the goal may change during the dialog as more information is captured. Goal estimation performance is important since it can facilitate the user achieving the correct action more quickly [7].

Y. Luan contributed to this work while doing internship at Merl. Current affiliation of Y. Luan is University of Washington

Intention understanding is framed as a semantic utterance classification problem while goal estimation is framed as classification problem of an entire dialog.

Conventional intention and goal estimation approaches use bag of word (BoW) features (or bag of intention features in goal estimation) as inputs to standard classification methods such as boosting, support vector machine, and logistic regression. However, one of the problems of applying BoW features to SLU tasks is that the feature vector tends to be very sparse. Each utterance only includes a dozen words at most (unlike document analysis). Therefore, the feature vector sometimes lacks enough semantic information to accurately estimate user intentions or goals. This paper investigates the use of additional semantic information by using word embedding [8] and latent topic model based on Latent Dirichlet Allocation (LDA) [9] techniques for intention understanding and goal estimation.

This paper first applies the semantic information as input features, and then uses them as an additional input layer in neural network architectures, by following the great success of neural network approaches in various SLU tasks. One of the most successful neural network approaches is based on Deep belief networks (DBNs) [10]. DBNs are stacks of Restricted Boltzmann Machines (RBMs), and their parameters are used as initial values when we estimate the neural network parameters by a back propagation algorithm (see [11] for more details). In the DBN context, the first step of finding initial parameters is called pre-training, and the second step of discriminative network training is called fine-tuning. Following the success of DBN/DNN training in ASR and image processing, researchers have applied other neural network architectures to SLU including DBN/DNN [12], Deep Convex Network [13], Recurrent Neural Network (RNN) [14, 15], and Long Short-Term Memory (LSTM) RNN [16].

However, in applying these techniques to SLU, one major difficulty is that we often have insufficient training data for a task, since the annotation of collected data can be expensive. The performance of a neural network trained in low resource conditions is usually inferior because of over-training. As mentioned above, our approach uses word embedding as an additional input layer of a neural network, which mitigates the over-training problem. To accomplish this, we initialize the affine transformation of the first layer by using a word embedding matrix estimated from a large-scale general corpus with unsupervised training methods (pre-training). Then, the entire SLU network is trained with the annotated training data, with word embeddings fine-tuned to the SLU task. This concept has been studied in [14, 17]. We also propose a novel Multi-scale RNN (MSRNN) architecture pre-trained with word embedding in order to capture long-term characteristics over the entire dialog for goal estimation. The proposed MSRNN uses two sub-networks

to model the different time scales represented by word and turn sequences. We applied a combination of pre-training and our proposed MSRNN network to a route guidance dialog task. The combination gives us a 18% relative error reduction compared to a baseline system.

2. Incorporating Semantic information

In this paper, we investigate the performance of importing semantic text embeddings to our SLU tasks. Two kinds of semantic features are trained in unsupervised fashion using large-scale web data. One is a word-level embedding which learns contextual information about word sequences by a feed-forward neural network frame. The other is document-level topic embedding, which models the latent topic information of a given sentence or article. The two kinds of semantic features have respective advantages: word-level embedding captures local contextual information, while topic embedding capture the statistics of the corpus, and thus provides more global information.

2.1. Word embedding

Many current NLP systems use a bag-of-words or one-hot word vector as an input, which leads to feature vectors of extremely large dimension. An alternative is a word embedding, which projects the large sparse word feature vector into a low-dimensional, dense vector representation.

There are two main model families for learning word vectors: 1) matrix factorization methods, such as latent semantic analysis (LSA) [18] and LR-MVL [19] and 2) neural network language model (NNLM) based methods, which model on local context window, such as Continuous Bag of Words (CBOW), Skip-gram [8] and others [20, 21]. Most word vector methods rely on the distance or angle between pairs of word vectors as the primary method for evaluating the intrinsic quality of word representations. Mikolov et al. [22] introduced an evaluation scheme based on word analogies, which favors models that produce dimensions of meaning. Among all the methods, CBOW and Skip-gram are the current state-of-the-art for the word analogy task. CBOW predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word. Mikolov's toolkit 'word2vec' which implement Skip-gram and CBOW can train on large-scale corpora very efficiently. Therefore, in this paper, we used word2vec to train Skip-gram and CBOW on a large scale web corpus collected from the internet and chose the embedding that gave the best SLU performance.

The typical word usage in our route guidance task differs from web texts, so we expected that fine-tuning of the word embedding would yield additional performance improvements by adapting the word embedding model to our target task.

2.2. Latent topic models

Latent topic models are algorithms that can discover semantic information from a collection of documents. Topic embedding, widely used in information retrieval, treats a document as a mixture of topics and uses a vector to represent the topic distribution. Classic latent topic models that have been used for SLU include Probabilistic Latent Semantic Analysis (PLSA) [23], LDA [9], Correlated Topic Model (CTM) [24] and Pachinko Allocation Model (PAM) [25]) all of which use Bayesian inference to compute the distribution of latent topics. Most latent variable models are generative models, which therefore can be used in unsupervised fashion.

Our framework uses LDA, which models each sentence or document as a mixture of proportions for latent components by using a Dirichlet prior. LDA has great performance on large-scale corpus and can be trained efficiently [26]. However, since LDA embedding is obtained an iterative inference algorithm (variational EM) or sampling method, it is hard to fine-tune an LDA embedding within a neural network framework. Therefore, in this paper, we do not fine-tune LDA features. It could be possible to use a deep unfolding method to fine-tune the LDA inference, such as that presented in [27].

3. Fine-tuning of linear input networks

The baseline method uses a discriminative approach to represent the goal and intention estimation models, and we can flexibly incorporate various information via feature engineering. We use multivariate logistic regression to compute $P(g|X)$ for classification target g and feature vector X as

$$P(g|X) = \text{softmax}([WX]_g) \tag{1}$$

where $[Y]_g$ means a g-th raw element of vector Y. The softmax function is defined as follows

$$\text{softmax}(z_m) \triangleq \frac{e^{z_m}}{\sum_k \exp(e^{z_k})} \tag{2}$$

W is a weight matrix to be estimated at the training step. For intention prediction, X is a bag-of-words feature vector, and g is an intention category. For the goal estimation task, X is a bag-of-intentions including confidence scores for each predicted intention in the dialog history, and g is a goal category. The baseline model can be regarded as a shallow neural network, with only one input layer and one softmax output layer.

In order to import a word2vec embedding to the system, we begin by purely concatenating embedding X_w with baseline feature X_b, i.e.,

$$X = [X_b^\top, X_w^\top]^\top \tag{3}$$

Then for each turn or sentence, X_w is obtained by summing over normalized word2vec feature for each word in the turn or sentence:

$$X_w = \sum_{i \in \{1..T\}} \frac{X_w(i)}{||X_w(i)||} \tag{4}$$

where T is the number of words in the sentence or turn. $X_w(i)$ is word2vec feature for the i-th word in the input sequence pretrained by large web corpus.

Then we propose two structures of fine-tuning. One is a feed-forward structure, which we use to fine-tune the affine transformation obtained from the word2vec embedding. This is equal to adding a linear layer to the shallow baseline network. Since LDA uses the expectation-maximization (EM) algorithm to learn latent embeddings, it is hard to use an affine transformation to fine-tune LDA. Therefore we do not fine-tune the LDA features. Experiments shows that importing LDA features can improve performance on low-resource datasets.

The other method is to use MSRNN for different time scale inputs. We apply MSRNN to goal estimation which uses both the ASR result and the predicted intention as input. The affine transformation from the word2vec embedding can be fine-tuned during training of the RNN.

3.1. Feed-forward architecture

The feed-forward architecture changes the baseline structure by adding a linear hidden layer between the Bag-of-Words input

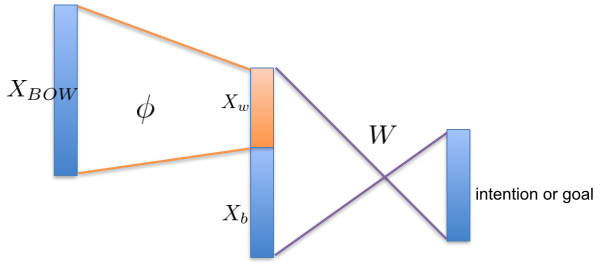

Figure 1: *Feed-forward architecture*

layer and the output layer (as Figure 1). The posterior probability of the goal/intention given the input features is calculated through softmax:

$$p(g|X) = \text{softmax}([W[X_I^\top, X_w^\top]^\top]_g) \quad (5)$$

where

$$X_w = \phi X_{BOW} \quad (6)$$

X_{BOW} is a Bag-of-Words vector of the input utterance with dimension of vocabulary size V. ϕ is a word embedding matrix initially learned from word2vec with dimensions $n \times V$, where n is the dimension of the word embedding. Eq. 6 is an affine transformation. W is the weight matrix between hidden layer and output layer. Fine-tuning is achieved by updating ϕ together with W. X_I is a vector with dimension of the number of intention categories, obtained by summing over the N-best intention confidence score. (The same X_I is used in the baseline method). We also tried a non-linear transformation using softmax function, but the result was not as good as the affine transformation.

The feed-forward architecture gives us flexibility in adjusting to the task domain, and in fact gives a better result than pure feature concatenation.

3.2. Multi-scale recurrent neural network (MSRNN)

As mentioned above, the goal estimation task has two input sequences for each sample: a word sequence and an intention sequence with confidence score. The two sequences have different time scales. However, the baseline architecture treats word input as bag-of-words, which ignores the contextual information of the input. Both input sequences, word and intention, contain contextual information, and intuitively a system that captures this information may perform better than one which does not. Therefore, we propose MSRNN architecture to model the different time scales represented by word and intention sequences, shown in Figure 2. The upper half of this figure represents the short time scale RNN, which accepts feature vectors for words in all history utterances, as an entire sequence. The lower half of the figure represents the long time scale RNN, which accepts a single intention feature vector for each utterance, and use all intentions in history as a sequence. The upper RNN has longer sequence than lower RNN. The two RNN structures have different parameters and are jointly trained. The goal is predicted at the end of each turn, the last layer of word sequence and last layer of intention sequence are concaternated to predict the output layer (goal of the current turn).

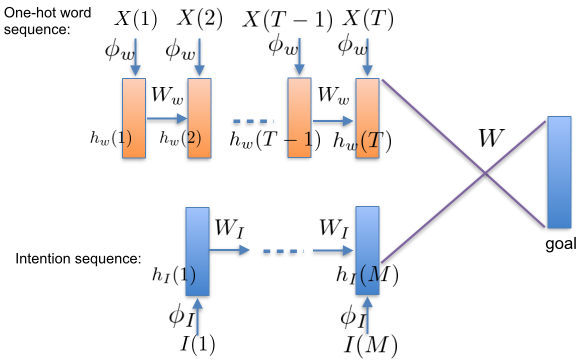

Figure 2: *MSRNN architecture*

This proposed architecture is formulated as follows:

$$p(g|\boldsymbol{X}, \boldsymbol{I}) = \text{softmax}([W[h_w(T)^\top, h_I(M)^\top]^\top]_g) \quad (7)$$

where, $\boldsymbol{X} = \{X(1), X(2), ..., X(T)\}$ and $\boldsymbol{I} = \{I(1), I(2), ..., I(M)\}$, T and M are the lengths of word sequence and intention sequence respectively. $X(t)$ is one-hot word vector. $I(m)$ is intention vector for m*th* utterance with N-best confidence scores on corresponding dimensions and the rest dimensions set to 0. $h_w(T)$ and $h_I(M)$ are the hidden activation vectors at T and M, which are explained below.

The recurrent module of word sequence and intention sequence can be calculated as:

$$h_w(t) = \text{sigmoid}(X(t)\phi_w + h_w(t-1)W_w) \quad (8)$$

$$h_I(m) = \text{sigmoid}(I(m)\phi_I + h_I(m-1)W_I) \quad (9)$$

and, we use the sigmoid function at the hidden layer defined as:

$$\text{sigmoid}(x) = \frac{1}{1+e^{-x}} \quad (10)$$

ϕ_w and ϕ_I are weight matrices between the raw input and the hidden nodes. ϕ_w is initialized by a word embedding matrix, and the back propagation through time is used to fine-tune ϕ_w. W_w and W_I are weight matrices between context nodes and hidden nodes. ϕ_I, W_w and W_I are randomly initialized.

4. Experiments

We built and tested models using data from a Japanese route guidance spoken dialog system. We used a 1.8G Japanese web text corpus from the Internet to train word2vec, and the full Japanese Wikipedia (900k documents) to train LDA. Combination of 50, 100, 150, and 300 word2vec dimensions, and 50, 100 LDA dimensions were tested. The dimensionality with the best result was selected. All text was tokenized and morpholized first by a Japanese morpholizer, chasen [28]. For each word, we use the word, pronunciation, and part of speech as individual token elements. Each token is in Word+Pronunciation+POS format. For both Japanese web data and Wikipedia data, low frequency words (frequency < 5) are replaced by UNKNOWN token. The total reduced vocabulary is around 150k.

4.1. Intention understanding

We evaluated the performance of fine-tuning on the intention understanding task. We collected a database of Japanese utterances in an automobile scenario with annotated intentions,

Table 1: Average error rate for intention prediction task, (ft) means with fine-tuning

	Dev (%)	Test (%)
Baseline (Bag-of-Words)	15.7	15.9
BoW + word2vec	13.0	13.5
BoW + word2vec (ft)	12.4	12.3

Table 2: Average error rate for goal estimation task, (ft) means with fine-tuning

	Dev (%)	Test (%)
Baseline (intention only)	18.9	19.5
intention (100 dim) + word2vec (50 dim)	17.2	17.5
intention (100 dim) + word2vec (50 dim + ft)	16.0	16.1
intention (100 dim) + word2vec (50 dim + ft) + LDA (50)	16.0	16.2
MSRNN	16.2	16.6
MSRNN (ft)	**15.6**	**15.9**

which contains 39,328 training samples, 5,000 dev samples and 5,000 test samples. The number of intention categories is 562, and the vocabulary size is 3,602. The text for intention understanding is first processed by slot filling. For example:

> Before slot filling: *Show flights from Boston to New York today*
>
> After slot filling: *Show flights from <City-departure> to <City-arrival> today*
>
> Intention: find_flight

Boston is replaced by <City-departure> and New York is replaced by <City-arrival>. In the baseline system, slots are treated equally as words. Each sentence is represented as a Bag-of-Words feature vector. We tested the performance of word2vec features by concatenating the 300-dimensional word2vec features with the Bag-of-Words features as described in Equations 3 and 4, and additionally tested the system both with and without fine-tuning.

The experimental results are shown in Table 1. By using the word2vec features, the performance was improved from the baseline system by 2.7% (dev.) and 2.4% (eval) (absolute reduction in error). With the fine-tuning, the performance further improves to 3.3% (dev.) and 3.6% (eval), absolutely. This result confirms the effectiveness of fine-tuning for our intention understanding task.

4.2. Goal estimation

We collected a dataset from a Japanese rule-based spoken dialog system, which contain 7059 turns in total. The log data contains the Japanese ASR results, system prompts, estimated N-best intentions, system actions, and an annotated goal for each turn. The intention vector consists of a sparse vector concaternated by N-best intentions (545 dimensions) and N-best system actions (545 dimensions). The dimension for user intention and system action together is 1090, the dimension for goal is 544 in total. Since the data size was very limited, we split the data into 4 folds, $3/4$ used for training, $1/8$ for validation and $1/8$ for test respectively. To evaluate the performance of our proposed approaches, we used the average accuracy across the 4 folds. We compared the result of using embeddings alone versus embeddings with fine-tuning. In the RNN architecture, hidden layer sizes of 100, 200 and 300 for the intention RNN module were trained and the one with best result was selected (100 for both with and without fine tuning). The model was trained by stochastic gradient descent.

We observe from Table 2 that importing semantic text features gives a better performance than using intention features alone. The best performance occurs on smaller semantic embedding dimensions and smaller hidden layer dimensions. The reason for this is due to the limited data size, for which fine-tuning high dimensional embeddings could lead to over-training problems.

Fine-tuning word embeddings improves upon the feature engineering results by a small but consistent margin. The first four rows of the table are implemented by the feed-forward structure and the last two rows are implemented by the MSRNN structure. Among all the feed-forward structures, fine-tuning word2vec plus LDA features gives the best result with 2.9% (dev.) and 3.3% (eval.) absolute improvement from the baseline. The MSRNN structure itself (without fine-tuning) already gives a significant performance improvement over simply importing semantic text features. When we add fine-tuning to the MSRNN training, we get the best overall result of 3.3% (dev.) and 3.6% (eval.) absolute improvement from the baseline. This proves that modeling time sequential input by using our MSRNN architecture with different time scales gives a gain to the system.

5. Conclusion

We propose an architecture for efficient learning for low resource SLU tasks. We pre-train a word embedding in an unsupervised way and fine-tune it for our specific SLU task. In order to capture long-term characteristics over an entire dialog, we implement a novel Multi-scale RNN structure which uses two sub-networks to model different time scales represented by word and intention sequences. We evaluated the performance of importing semantic features, fine-tuning by feed-forward neural network and MSRNN structure. All three methods improve SLU accuracy. The MSRNN architecture outperforms the feed-forward structure and gives the best overall result which improves the goal estimation baseline by 3.6%. The first part of future work will focus on extending the proposed neural network architecture to deal with more advanced recurrent architecture, e.g., LSTM [16, 29] to capture the sequential information in the dialog more precisely. Then sentence level embeddings that are successfully used in sentiment and topic classification will be used and adapted to our SLU task.

6. References

[1] D. Jurafsky and J. H. Martin, *Speech & Language Processing*. Pearson Education, 2000.

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

[3] A. L. Gorin, G. Riccardi, and J. H. Wright, "How may I help you?" *Speech communication*, vol. 23, no. 1, pp. 113–127, 1997.

[4] V. Zue, S. Seneff, J. R. Glass, J. Polifroni, C. Pao, T. J. Hazen, and L. Hetherington, "JUPITER: a telephone-based conversational interface for weather information," *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 85–96, 2000.

[5] R. Pieraccini and J. Huerta, "Where do we go from here? Research and commercial spoken dialog systems," in *Proceedings of the 6th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2005.

[6] D. Bohus and A. I. Rudnicky, "The RavenClaw dialog management framework: Architecture and systems," *Computer Speech & Language*, vol. 23, no. 3, pp. 332–361, 2009.

[7] S. Watanabe, J. R. Hershey, T. K. Marks, Y. Fujii, and Y. Koji, "Cost-level integration of statistical and rule-based dialog managers," in *INTERSPEECH*, 2014, pp. 323–327.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[9] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[10] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[11] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[12] R. Sarikaya, G. E. Hinton, and A. Deoras, "Application of deep belief networks for natural language understanding," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 22, no. 4, pp. 778–784, 2014.

[13] L. Deng and D. Yu, "Deep convex net: A scalable architecture for speech pattern classification," in *Proceedings of the Interspeech*, 2011.

[14] G. Mesnil, X. He, L. Deng, and Y. Bengio, "Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding," in *Interspeech 2013*, August 2013.

[15] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Gao, "Recurrent conditional random field for language understanding," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4077–4081.

[16] K. Yao, B. Peng, Y. Zhang, D. Yu, G. Zweig, and Y. Shi, "Spoken language understanding using long short-term memory neural networks," in *Interspeech 2013*, December 2014.

[17] X. Song, X. He, J. Gao, and L. Deng, "Unsupervised learning of word semantic embedding using the deep structured semantic model," Tech. Rep. MSR-TR-2014-109, August 2014.

[18] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis," *JAsIs*, vol. 41, no. 6, pp. 391–407, 1990.

[19] P. Dhillon, D. P. Foster, and L. H. Ungar, "Multi-view learning of word embeddings via cca," in *Advances in Neural Information Processing Systems*, 2011, pp. 199–207.

[20] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 160–167.

[21] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: a simple and general method for semi-supervised learning," in *Proceedings of the 48th annual meeting of the association for computational linguistics*. Association for Computational Linguistics, 2010, pp. 384–394.

[22] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations." in *HLT-NAACL*, 2013, pp. 746–751.

[23] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 50–57.

[24] D. M. Blei and J. D. Lafferty, "A correlated topic model of science," *The Annals of Applied Statistics*, pp. 17–35, 2007.

[25] W. Li and A. McCallum, "Pachinko allocation: Dag-structured mixture models of topic correlations," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 577–584.

[26] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent Dirichlet allocation," in *advances in neural information processing systems*, 2010, pp. 856–864.

[27] J. R. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *arXiv preprint arXiv:1409.2574*, 2014.

[28] Chasen. [Online]. Available: http://chasen.naist.jp/hiki/ChaSen/

[29] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, 2014, pp. 3104–3112.