Multi-task Learning of Spoken Language Understanding by Integrating N-Best Hypotheses with Hierarchical Attention

Mingda Li Xinyue Liu Weitong Ruan Luca Soldaini Wael Hamza Chengwei Su
Amazon Alexa AI, Cambridge, USA
{mingda, luxnyu, weiton, lssoldai, waelhamz, chengwes}@amazon.com

Abstract

Currently, in spoken language understanding (SLU) systems, the automatic speech recognition (ASR) module produces multiple interpretations (or hypotheses) for the input audio signal and the natural language understanding (NLU) module takes the one with the highest confidence score for domain or intent classification. However, the interpretations can be noisy, and solely relying on one interpretation can cause information loss. To address the problem, many research works attempt to rerank the interpretations for a better choice while some recent works get better performance by integrating all the hypotheses during prediction. In this paper, we follow the way of integrating hypotheses but strengthen the training mode by involving more tasks, some of which may be not in existing tasks of NLU but relevant, via multi-task learning or transfer learning. Moreover, we propose the Hierarchical Attention Mechanism (HAM) to further improve the performance with the acoustic-model features like confidence scores, which are ignored in the current hypotheses integration models. The experimental results show that compared to the standard estimation with one hypothesis, the multi-task learning with HAM can improve the domain and intent classification by relatively 19% and 37%, which are much higher than improvements with current integration or reranking methods. To illustrate the cause of improvements brought by our model, we decode the hidden representations of some utterance examples and compare the generated texts with hypotheses and transcripts. The comparison shows that our model could recover the transcription by integrating the fragmented information among hypotheses and identifying the frequent error patterns of the ASR module, and even rewrite the query for a better understanding, which reveals the characteristic of multi-task learning of broadcasting knowledge.

1 Introduction

In an SLU system (Tur and De Mori, 2011), the domains and intents are usually inferred by natural language understanding (NLU) modules with the hypotheses mapped from input speech by ASR module. For each speech audio, the transferred hypothesis is the one with the highest recognition score. However, due to the unsatisfactory ASR accuracy (Xiong et al., 2018; Barker et al., 2018), the 1-best hypothesis may contain errors. To solve the problem, there are some research works rescoring (reranking) the n-best hypotheses[1] to reduce the word error rate (WER) by dual comparison with a discriminative language model (Ogawa et al., 2018; Ogawa et al., 2019); or involving morphological, lexical, syntactic or confidence score features for reranking (Sak et al., 2011; Collins et al., 2005; Chan and Woodland, 2004; Peng et al., 2013; Morbini et al., 2012).

In contrast to the reranking models, which predict only one hypothesis with the lowest WER and transfer that hypothesis to NLU modules, there is recently another attempt to integrate the fragmented information among the n-best hypotheses by feeding all the hypotheses together to NLU modules (Li et al., 2020; Li, 2020). The proposed approaches to integrating hypotheses include hypothesized text

[1]We use ASR *n-best hypotheses* or n-bests to denote the top n interpretations of a speech and the 1-best or 5-best stands for the top 1 or 5 hypotheses. The hypotheses are ranked by the associated confidence scores.

Proceedings of the 28th International Conference on Computational Linguistics: Industry Track, pages 113–123
Barcelona, Spain (Online), December 12, 2020

concatenation (Combined Sentence) and hypotheses embedding concatenation (PoolingAvg and PoolingMax). Compared to the accuracy on the oracle reranking results (i.e., picking the hypothesis most similar to transcription), the PoolingAvg achieves much higher improvements for the NLU tasks. However, the integration framework can be further improved by introducing more tasks with Multi-Task Learning (MTL) or Transfer Learning and involving more features to optimize the integration process.

MTL (Zhang and Yang, 2017; Liu et al., 2019; Caruana, 1997) is a widely used machine learning paradigm for simultaneously training related tasks. In the MTL training, one task can apply the knowledge learned from others. MTL can improve the generalization of the trained model by avoiding overfitting to a single task and make full use of all the labeled data from all tasks to solve the issue of insufficient training data. The MTL has been shown efficient for some natural language processing tasks outside the SLU system like text similarity, pairwise text classification (Liu et al., 2019). In contrast to multi-task learning, by transfer learning or domain adaption (Pan and Yang, 2009; Howard and Ruder, 2018; Torrey and Shavlik, 2010), some tasks (source tasks) can be trained in the first stage knowing nothing about the other tasks (target tasks). While in the second stage, the embeddings from pre-trained model are fine-tuned according to the target mission. The transfer learning cares more about the target tasks. Some popular fine-tunable pre-trained models like BERT (Devlin et al., 2018), ELMO (Peters et al., 2018) nail down the transfer learning in NLP.

The rest of the paper is organized as follows. Sec. 2 presents various models and training paradigms explored in this work. Sec. 3 describes our experimental details, results and analysis. Sec. 4 concludes our findings and discusses the future directions.

2 Models

We start by reviewing different categories of SLU system designs in Sec. 2.1. Those designs have achieved great success, but they are trained solely on one task and cannot borrow the knowledge from some relevant tasks like transcription reconstruction. To involve more tasks during training, we explore two paradigms to: 1) train them simultaneously in a single stage (Sec. 2.2), which is actually multi-task learning; 2) train them asynchronously (Sec. 2.3) in multiple stages, which includes two ways of using the pre-trained model from the first stage (transfer learning or text generation). In Sec. 2.4, we illustrate the importance of acoustic-model features and the way of utilizing them hierarchically.

2.1 The Standard SLU, Reranking, Integration And Oracle

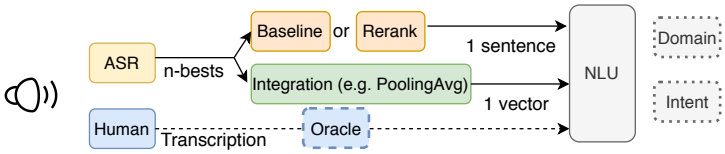

Figure 1: The pipelines of current SLU systems with various ways to exploit hypotheses (with the Oracle).

We firstly review the current designs for the SLU system in Figure 1, which include the standard pipeline (Baseline), Reranking and Integration models. In production, the input audio is transcribed by ASR to get n-best hypotheses. Then, the Baseline model will take the one with the highest confidence score for NLU tasks. Nevertheless, the Reranking models do not solely rely on the confidence scores generated by the ASR module. They prefer to rescore the interpretations based on more features like semantic information and choose the most reliable one. Both Baseline and Reranking models transfer one sentence to the NLU module for classification. However, some recent works like (Li et al., 2020) indicate this causes information loss and attempt to use all the hypotheses during classification. They embed each hypothesis to one vector and unify the vectors to one by a pooling layer, which becomes the input to the NLU task. Ideally, we can make the hypothesis close to the transcribed sentence by humans. To know the ceiling point of performance, there is always the Oracle model predicting with human transcriptions.

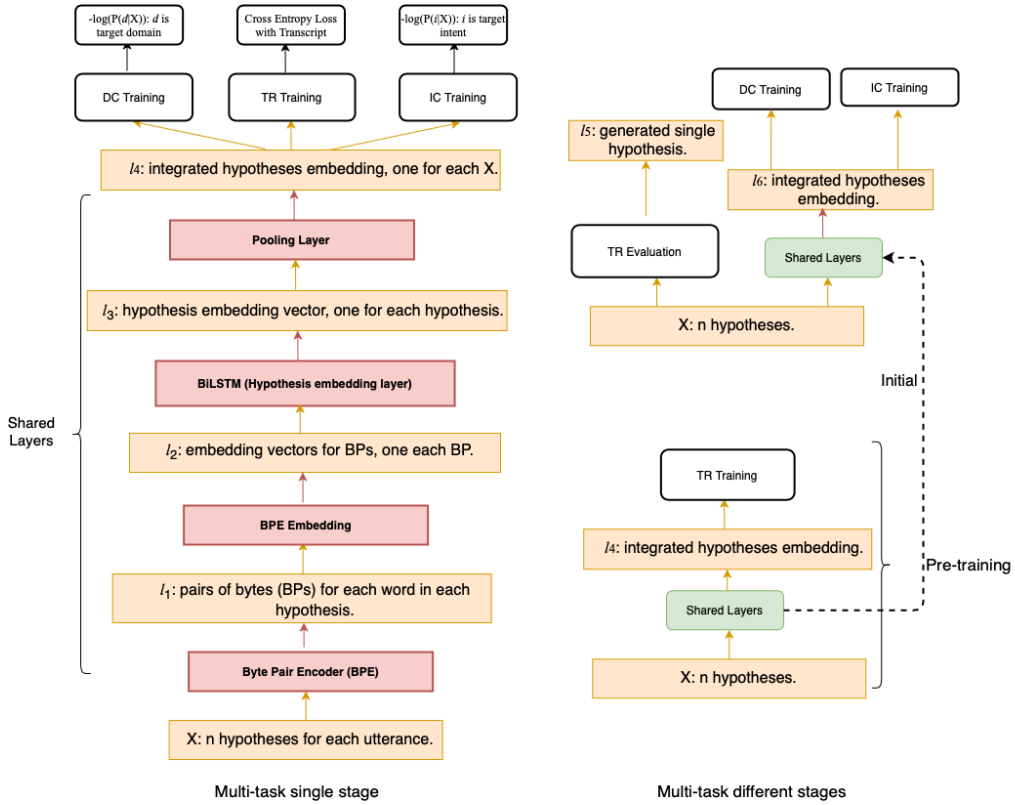

Figure 2: The architecture of multi-task training in a single stage or different stages. The left side is training all tasks (TR, DC, IC) in the same stage while the right side is to train TR firstly and fine-tune or generate texts base on the pre-trained TR model for DC and IC.

2.2 Multi-task Learning: Training Tasks Simultaneously in A Single Stage (MT_S)

Although the current approaches have gained a large improvement, their training is exclusive for one task each time and overlooks the knowledge from other tasks, which can be improved by considering more relevant tasks simultaneously with MTL. The left side of Figure 2 shows the design of training multiple tasks (transcription reconstruction, domain classification and intent classification) simultaneously for integrating n-bests. The lower layers are shared and the top two layers are task-specific.

Shared Layers: In shared layers, the input $X = \{x_1, ...x_n\}$, are n hypotheses generated by the ASR module for one speech. To decrease the embedded vocabulary size, the hypothesis is split to subword units (byte of pairs or BPs) in l_1 by a byte pair encoder (Sennrich et al., 2015) and each BP is embedded to a vector in l_2. Then, the BiLSTM encoder gets contextualized representations for the BPs ($bp_{H_i,1}...bp_{H_i,j}$) of the hypothesis (H_i) containing j byte pairs as follow:

$$(h_{H_i,1}, ..., h_{H_i,j}) \leftarrow BiLSTM_\theta(bp_{H_i,1}, ..., bp_{H_i,j}). \quad (1)$$

Each hidden state is the concatenation of the forward and backward directions, e.g. $[h_{H_i,1f}, h_{H_i,1b}]$, where f means forward and b means backward. The finally utilized output state for H_i is the concatenation of the last hidden state of the forward and backward LSTM, i.e. $h_{output_i} = [h_{H_i,1b}, h_{H_i,jf}]$. To integrate the output states of all hypotheses, we follow the empirically best approach in Li et al. (2020), PoolingAvg, which firstly pads into the output state of the first best hypothesis by $n - r$ times when the amount of hypotheses, r is smaller than n. Then, a unified representation $h_{unified}$ can be achieved by average pooling (n by 1 sliding window and stride 1) for the n output states in layer l_4. In the PoolingAvg, the unified representation is used to predict the domain or intent and all the parameters are trained by the cross entropy loss for the classification task. While in our method, we introduce a new task and train tasks simultaneously. Below, we discuss the task specific layers and the training objective.

Transcription Reconstruction (TR): For all the natural language understanding tasks, it is important to obtain a high-quality unified representation of the incoming utterance. To assure the quality of $h_{unified}$,

we consider the task to reconstruct transcription by an LSTM decoder adopting $h_{unified}$ as the initial state of its first recurrent layer. Once the decoder's output is close to transcription, it shows the representation contains the high-quality information of transcription. The task is trained based on cross entropy loss:

$$\mathcal{L}_{CE-TR} = \sum_{bp=1}^{|S|} \sum_{e=1}^{|V|} y_{bp,e} log(1/\hat{y}_{bp,e}). \tag{2}$$

The S is the transcription while the bp represents the bp^{th} byte pair inside S. The e represents the e^{th} byte pair in the vocabulary. Each time, $y_{bp,e}$ is 1 when the bp^{th} byte pair is the e^{th} entry of vocabulary and 0 otherwise. $\hat{y}_{bp,e}$ is the predicted probability that the e^{th} byte pair should appear at bp^{th} position. With the transcription reconstruction, the model can learn some erroneous patterns between the n best hypotheses and target transcriptions and recover accordingly. For example, one phrase may always be mis-recognized as another phrase by an ASR module. During our evaluation, with a set of utterance examples, we decode the hidden states and show the recovering capability.

Domain Classification (DC): With the same output hidden state, we could as well predict the domains (e.g. music, weather or knowledge) by a multilayer perceptron (MLP) (Mather and Tso, 2016) module. The loss for the DC task is:

$$\mathcal{L}_{CE-DC} = \sum_{d=1}^{|D|} y_{u,d} log(1/\hat{y}_{u,d}). \tag{3}$$

The $y_{u,d}$ is the indicator function which equals to 1 when the utterance belongs to the d^{th} domain of the candidates set D. The $\hat{y}_{u,d}$ is the predicated probability, $\hat{y}_{u,d} = softmax(f_{MLP-DC}(h_{unified}))$, where the f_{MLP-DC} contains the parameters to be trained in DC task.

Intent Classification (IC): Then, we could further utilize $h_{unified}$ for domain-specific intent prediction with another MLP module. For an incoming utterance, it is usually firstly classified to one domain and the intent classification will be domain-specific (Tur and De Mori, 2011). The loss of the IC task is:

$$\mathcal{L}_{CE-IC} = \sum_{i=1}^{|I|} y_{u,i} log(1/\hat{y}_{u,i}), \tag{4}$$

where the $y_{u,i}$ is 1 when the utterance should be classified to the i^{th} intent. The $\hat{y}_{u,i} = \text{softmax}(f_{MLP-IC}(h_{\text{unified}}))$, where $\hat{y}_{u,i}$ is the predicted probability of the utterance belonging to the i^{th} intent and f_{MLP-IC} contains the task-specific parameters.

Training Objective: For the PoolingAvg method, the objective is to minimize the $\mathcal{L}_{CE-IC}$ or $\mathcal{L}_{CE-DC}$, while for our MTL framework, the objective is minimizing

$$\mathcal{L} = \lambda_{TR}\mathcal{L}_{CE-TR}/|S| + \lambda_{IC}\mathcal{L}_{CE-IC} + \lambda_{DC}\mathcal{L}_{CE-DC}, \tag{5}$$

where the λ_{TR}, λ_{DC} and λ_{IC} are the weights of the loss functions associated with corresponding tasks. Since for one utterance, the target transcription contains multiple words and the $\mathcal{L}_{CE-TR}$ is the sum of loss for all the words, we utilize the normalized version of the transcription reconstruction loss.

The Language Model: During our experiments, we also tried the multi-layer transformer (Vaswani et al., 2017) for the encoder. We find it costs more for training or evaluation while bringing no improvements. In addition, since the length of hypotheses varies, it is hard to align variant-length output states of different n-best hypotheses and exploit the attention between encoder and decoder (if we also use the Transformer's decoder) for the TR task. Thus, we exploit the BiLSTM encoder and LSTM decoder.

2.3 Multi-task Training in Multiple Stages (MT_M) with Transfer Learning or Text Generation

Another way to train the above-mentioned tasks is in different stages as shown in the right part of Figure 2. Inasmuch as for all the NLU tasks, it is necessary to obtain a high-quality hypothesis representation. We prioritize the training of TR in the first step and let all NLU models share the same pre-trained TR model. The approaches of exploiting the pre-trained model are introduced as follows.

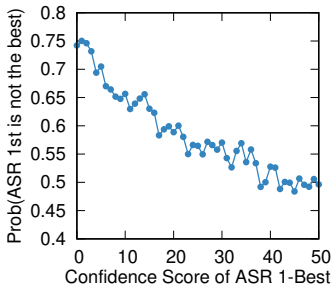

Figure 3: Confidence score of ASR 1-Best (bin size 10‰).

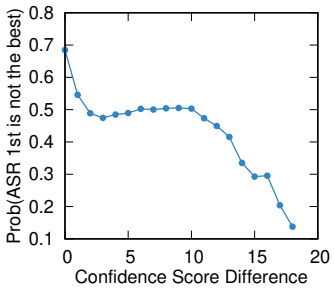

Figure 4: Confidence score difference (bin size 5‰).

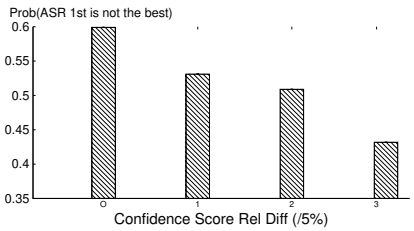

Figure 5: Confidence score relative difference (bin size 5%)

Transfer Learning: One way to use the TR task from the first step is transfer learning, where we regard the TR as the pre-training step and let the DC, IC tasks adapt the knowledge by fine-tuning. We call the method following this idea as Transfer Learning (TL). The parameters of the pre-trained TR's shared layers, including the embedding of byte pairs and the BiLSTM encoder, are used as the initial value. The TR task-specific parameters like the decoder part's are discarded. Then, the shared layers' parameters and task-specific layer's parameters, in f_{MLP-DC} or f_{MLP-IC}, are all trained during the fine-tuning step. Although DC and IC share the same initialization parameters, their fine-tuned models are separate. The benefit of the two-step training is that the model and knowledge from pre-training step can serve multiple downstream tasks. In addition, with the well-initialized parameters, it saves much time for fine-tuning.

MT_M with Text Generation: Since the TR model has been tuned to recover the errors contained in the ASR n-bests, we can firstly evaluate it to generate the text closer to transcription. Then, the domain or intent can be predicted based on the generated text. This method is called Multi-task Multi-stage with Text Generation (MMTG). At this moment, the input to DC or IC is only one generated hypothesis instead of n-bests. To predict with one hypothesis, we can exploit the IC or DC models pre-trained on transcription or 1-best, which only expect one sentence as the input.

2.4 Hierarchical Attention on Byte Pair Embedding and Hypothesis Integration Layer

All the above algorithms treat the input hypotheses as normal natural language to process but ignore that the hypotheses are generated by ASR and associated with more acoustic-model information than the text itself. For example, the position information (whether the hypothesis is the first best or the last best), the difference of confidence score associated with the first best and second best hypothesis, etc. The acoustic-model features have been proven to be valuable for many applications including: 1) the arbitration task to select the best among client and service recognition results (Kumar et al., 2015), 2) the Recognizer Output Voting Error Reduction (ROVER) (Fiscus, 1997), which takes the outputs generated by multiple ASR systems to generate one output with reduced error rate, 3) confidence normalization (Kumar et al., 2014), etc. In this section, we would like to introduce those features, why they can be helpful, and how they can hierarchically take part in the shared layers in the left side of Figure 2.

2.4.1 Acoustic-model Information

Those features can be divided into three categories including confidence-score features, positional information and confusibility. We illustrate their close relationship with the hypothesis quality as following.

(a) Confidence-score Features: The confidence scores quantitatively represent the correctness of recognized hypotheses and words in a [0‰,1000‰] range. Plenty of previous research works have proven the effectiveness of those features. Here, we take the confidence score of hypothesis as an example to show the valuable information contained in confidence scores. For each utterance in the training set, we evaluate the probability that ASR 1-best is not the best for different scales of the ASR 1-best confidence score. A hypothesis is the best when it is the most similar one to the transcription considering the edit distance. In Figure 3, it is obvious that a higher confidence score means a higher probability that the ASR 1-best is the best one among hypotheses.

(b) Positional Information: The ranking position of the hypothesis is another important information. To show its significance, we gain the distribution of exact matchings, i.e. the hypothesis is the same as the

transcription, between different ranking positions and the transcription. Among all the exact matching cases, 50% appear at the first best hypothesis while 19%, 13%, 10% and 6% occur at the 2^{nd}, 3^{rd}, 4^{th}, and 5^{th} best hypothesis. Hence, a more forward position does indicate a higher recognition quality.

(c) Confusibility: The features of the confusibility category include the difference ($confidence_{H_1} - confidence_{H_i}$) and relative difference ($difference/confidence_{H_1}$) of confidence score between the ASR 1-best and the others. The larger difference implies the lower confusibility to choose the first hypothesis as the best. As the confidence score of the first best should be larger or equal to the others, the difference and relative difference are non-negative values measuring the degree of outperforming. In the Figure 4 and 5, there is a trend that the larger difference (between ASR first and second best) implies the lower probability that ASR 1-best is not the best, which means it is easier to determine the ASR 1-best as the best. Here, we only show the difference to the second best as an example. While in later designs, the features will be formed based on the difference between each i-best's confidence score and the 1-best's.

2.4.2 Hierarchical Attention Mechanism (HAM)

We have shown that the acoustic-model information reveals the quality of recognition and to exploit them, we add them into shared layers hierarchically. The HAM is proposed by the hierarchical structure of the n-bests (BPs from a hypothesis, a hypothesis from n-bests). Similar hierarchical structures are realized in different areas, where various kinds of information like documentation (Yang et al., 2016), knowledge graphs (Hu et al., 2015), Internet network (Li et al., 2018), or voice queries (Rao et al., 2017) are encoded. While integrating n-bests, the process is building representation for one hypothesis from BPs and then aggregating them into an n-bests representation. We likewise exploit the acoustic-model information hierarchically to BPs embedding (HAM_BP) and then to the aggregation of hypotheses (HAM_H). The HAM_ALL exploits the information in both layers.

Byte Pair Embedding Layer (HAM_BP): In Figure 6 right side, we show the way of involving the byte pair acoustic-model information in the byte pair embedding layer of Figure 2 left side. Instead of concatenating the last hidden state of forward and backward LSTM for hypothesis embedding (Figure 6 left side), we would like to consider the quality of each byte pair and take into account the entire sequence of hidden states. To exploit the information, we firstly need to figure out the problem of missing acoustic-model information for byte pairs because we only have the confidence scores associated with words. Since it can be ensured that each byte pair only belongs to one word, we can assign the confidence score for a byte pair according to its parent, i.e. the word. For example, in Figure 6, the "low" and "-er" are two byte pairs of the word "lower", so they share the same confidence score of the word "lower", i.e. 0.9. To use the confidence scores as attention scores, we can normalize them by Softmax or convert them to bin value or logarithmic scale, etc (shown by $f(x)$ in Figure 6). The attention score matrix is multiplied with the hidden vectors matrix of the BiLSTM embedding, where each hidden vector concatenates its forward and backward states. The mean (from pooling) of weighted hidden state vectors forms a single vector for each hypothesis and the vector will participate in the following hypotheses integration layer.

Hypotheses Integration Layer (HAM_H): Figure 7 illustrates the integration of hypotheses with their associated acoustic-model information. The left side of the dotted line in Figure 7 is the way to integrate without the acoustic-model features in Figure 2 layer l_4, where hypotheses embeddings transferred from the l_3 layer are combined by the pooling operation. The problem is that the normal pooling layer treats hypotheses equally, although the quality of the n-best hypotheses actually varies. We add a Multiple Layer Perceptron (MLP), i.e. Feedforward (FFW), to synthesize all the features revealing the quality of each hypothesis, including the positional information, confidence score, confidence score difference and confidence score relative difference. The output of FFW is normalized via Softmax and works as the attention scores or weights. In MatMul, we multiply the attention score matrix and the hypothesis embedding matrix. Finally, the weighted embeddings are combined through pooling.

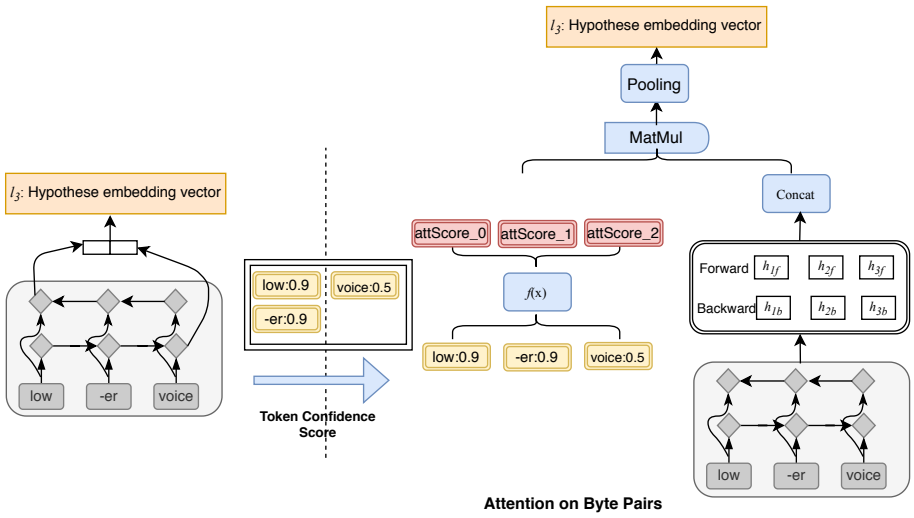

Figure 6: Attention on byte pair embedding.

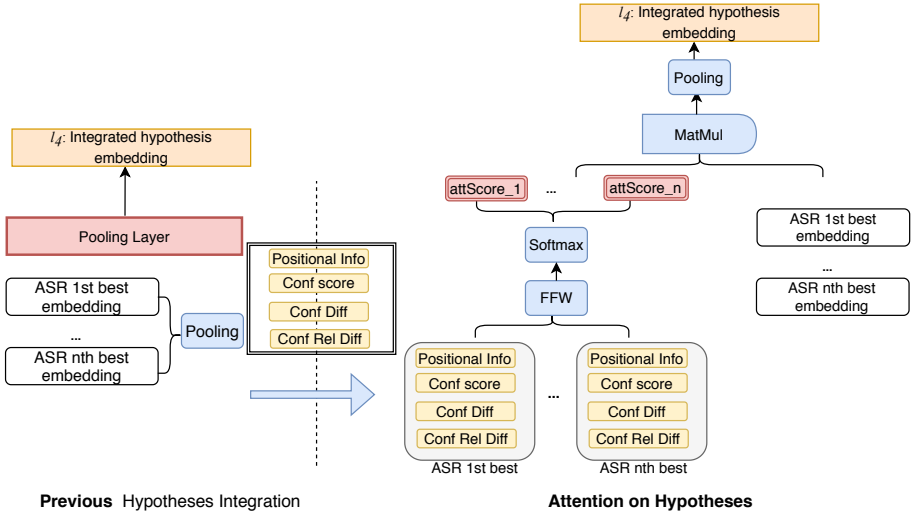

Figure 7: Attention on hypotheses integration.

3 Experiments

3.1 Dataset and Models

Our data consists of ~ 9M anonymized English utterances. The utterances are divided into training, development and testing parts with 8:1:1 ratio. They are annotated with 23 domains and further classified into different intents for each domain. The transcripts are hand-transcribed by humans.

The compared approaches include the Baseline model and Oracle model mentioned in Sec. 2.1, PoolingAvg, Oracle of Reranking Model and the approaches mentioned in this paper. The PoolingAvg is the foremost one among all the models integrating n-best hypotheses (Li et al., 2020). The Oracle of Reranking Model makes prediction by the hypothesis most similar to the transcription each time. As for the models in this paper, they include the multi-task training in a single stage (MTL, Sec. 2.2), or in different stages (TL and MMTG, Sec. 2.3) and the HAM (Sec. 2.4). The HAM actually modifies the shared layers and is not task-specific, so it is possible to combine it with the MTL or transfer learning mechanism. For example, MTL_{HAM} means using HAM to modify shared layers and training with mode of MTL. For all the models, the byte pairs are embedded to a 128-dimensional space. The hidden states in the BiLSTM encoder of the shared layers and the LSTM decoder of task TR are both 512-dimensional. The training iterator is a fixed mini-batch iterator with size 128 and each model is trained for ten epochs, while the model providing the highest performance for the development data is selected.

3.2 Domain Classification

		Entire test set	Agree Part	Disagree Part
Category	Model	RErr(%)	RErr(%)	RErr(%)
Standard	Baseline	0.00	0.00	0.00
Rerank	Oracle	3.71	0.00	7.25
Integration	PoolingAvg	14.29	3.56	24.67
MT_S	**MTL**	**17.99**	**7.67**	**28.26**
MT_M	TL	17.34	7.32	27.37
	MMTG	9.16	1.88	16.20
HAM	MTL_{HAM_H}	18.10	7.88	28.22
	MTL_{HAM_BP}	18.10	8.07	28.05
	MTL_{HAM_ALL}	**19.30**	**9.94**	**28.68**
Oracle		27.04	0.00	53.02

Table 1: Relative error reduction (RErr) for domain classification.

Table 1 compares the domain classification performance of all the models. As seen from the results of the entire test set, the transfer learning, multi-task learning and the improved versions with HAM are all better than the existing methods. Among the ways of training multiple tasks synchronously or asychronously, the MTL works the best. In our experiments, we have tried different hyperparametric values in formula 5. Since the predictions of IC is based on the specific domain predicted in DC, we only have two tasks (DC, TR) and associated hyperparametrics $(\lambda_{DC}, \lambda_{TR})$ for MTL in the domain classification. We tried ratios of $\lambda_{TR} : \lambda_{DC}$ with $1 : i, i \in 1...10$. Here, we assign the weight for DC as a larger one because we care more about the DC task performance, while the TR is actually an auxillary task. Through experiments, we find the performance for ratio $1 : i, i \in 1...3$ is comparable to each other and better than the rests and we show the results for ratio $1 : 1$ in Table 1. With the acoustic-model information, the performance of MTL is further improved. Exploiting the acoustic-model information hierarchically on both hypotheses and byte pairs layers, i.e. HAM_ALL, is better than on one layer (HAM_H, HAM_BP).

To reveal the reason of improvements, we split the entire test set into two parts by whether the 1-bests agree with transcriptions or not and evaluate respectively. Comparing the Agree and Disagree part, we find that the gained improvements of models in MTL, TL and MTL with HAM mainly come from the disagreed part. This indicates that integrating more hypotheses could help more when 1-best differs from transcription. Later, we will illustrate the reasons more visually with some utterance examples.

3.3 Real Effect of MTL: Analysis of Utterance Examples

Domain Estimation Result			Generated Text (decoded hidden representation), ASR *n*-bests and transcription				
Baseline	MTL	Real	Transcription	Generated Text by MTL	ASR-1^{st}	ASR-2^{nd}	ASR-3^{rd}
Daily	Music	Music	play muse	play muse	play news	play muse	play mus
Knowledge	Video	Video	harry porter	harry porter	how do you porter	how do you patter	harry power
Communication	Help	Help	how the **call** service work	how the **remote** service work	how the call service work	how the cost service work	None

Table 2: Comparison among the decoded hidden representations, hypotheses and transcriptions.

We use the well-trained MTL on some utterance examples to predict their domains and decode the integrated hypotheses embedding (hidden representations) with Beam Search Decoder (beam size 1) to compare the generated text with the ASR hypotheses, transcription. In Table 2, the first three columns are the predicted domains of Baseline model , MTL model (predicted by the hidden representation) and the Real domain. The other columns compare the transcriptions, the decoder's generated text in MTL and the *n*-bests. Due to space limit, we only choose three typical example with their top 3 hypotheses. The number of hypotheses varies, so we use None for the missing one.

Why are we better than Baseline on Disagree Part? The reasons MTL can outperform the Baseline model's prediction on Disagree Part can be categorized into two as follows. 1) **Choose the best from ASR 2-*n* bests** (e.g. the third row in Table 2): In this condition, there is a high-quality hypothesis ("play

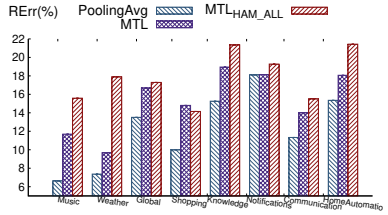

Figure 8: Improvements on 8 important domains.

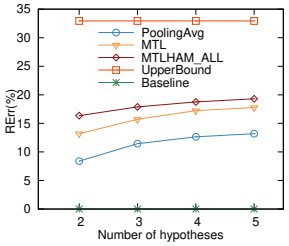

Figure 9: Comparison under different amount of n-bests.

Domain	Shopping	Knowledge	Comm
Baseline	0.0	0.0	0.0
Oracle	47.63	40.28	32.89
PoolingAvg	25.55	25.00	11.92
MTL	35.02	36.11	12.36
MTL$_{HAM_ALL}$	36.91	30.56	12.58

Table 3: Intent classification: relative error reduction versus Baseline.

muse") within the n-bests. The position of the hypothesis is not the first one but we can correctly identify it in the generated text. 2) **Integrate fragmented information** (e.g. the forth row in Table 2): In this condition, the transcription "harry porter" spread out over hypotheses. The "harry" is in the third best while the "porter" is in the first best. We can collect the information and recover the "harry porter" in the generated text. The ability of integration can thus been shown. The ability can be obtained by learning the error patterns between ASR hypotheses and transcriptions during the TR task.

Why are we even better than Baseline on Agree Part? The transcriptions should be the golden information but we can still outperform the Baseline's prediction from ASR 1-best when ASR 1-best agrees with the transcription. The reason is **Query rewriting** (e.g. the fifth row): We find the trained model attempt to rewrite transcription when it may cause misunderstanding. In the fifth row, while the transcription is "how the call service work", the trained model replaces the sensitive word "call" with another word "remote" with similar meaning or embedding position. The word "call" is a sensitive word because it always occurs in the Communication domain, which can make the predictor mis-classify it. However, the word "remote" is not sensitive but semantically similar to the word "call". This example is also a perfect demo to show the effect of multi-task learning. The multi-task learning here is to find the balance point between the domain classification and transcription reconstruction. Considering both tasks will propel the model to rewrite a query with a similar semantic meaning and avoid misunderstanding.

We summarize the different causes of improvements by some utterance examples here to offer an insight of the model's real effect. However, we do not show the numerical analysis like WER because it is hard to evaluate whether the decode generation is high-quality considering the query rewriting.

3.4 Improvements on Different Domains and Different Numbers of Hypotheses

Now, we compare more specifically the MTL, MTL$_{HAM_ALL}$ and PoolingAvg on 8 important domains out of the whole 23 domains in Figure 8. The performance of each of the three models will be compared to the baseline model and the relative error reduction (RErr) is shown. This result shows that the MTL gains more improvements than the best integration model PoolingAvg for all the 8 domains while the HAM can enhance the performance on almost all 8 domains (except an acceptable decay for Shopping).

All the previous results of models based on n-best actually utilize 5-best hypotheses and we also want to see the performance with different number of hypotheses. In Figure 9, we could find the best model is always MTL$_{HAM_ALL}$ for different numbers of utilized hypotheses. There is also a trend that after 4 hypotheses are utilized the growth become more gentle. The lines for Baseline and UpperBound are flatten because they are only based on ASR 1-bests and transcriptions. We only show the performance until 5 hypotheses are utilized because: 1) Most of our ASR recognition results only contain at most 5-bests; 2) In production, the more hypotheses are utilized, the slower it will be for training and testing. We only want to afford up to 5 hypotheses considering response delay.

3.5 Intent Classification on Three Important Domains

Another task, intent classification, is domain specific and we show the IC of 3 important domains. Table 3 shows the relative error reduction compared to Baseline model. The multi-task learning for intent classification considers both the intent classification and transcription reconstruction. The result showed is under the loss ratio $\lambda_{TR} : \lambda_{IC} = 1{:}1$ for the two tasks. We can find the MTL$_{HAM_ALL}$ and MTL outperforms the foremost PoolingAvg for all three domains' domain-specific intent classification.

4 Conclusion and Future Work

This work is motivated by introducing multi-task learning (MTL), transfer learning (TL) and acoustic-model information into the framework of integrating n-best hypotheses for spoken language understanding. Among those algorithms, we find the MTL results in higher performance compared to the TL. For the acoustic-model information, we illustrate their close relationship with the hypothesis quality and utilize the hierarchical attention mechanism to include the information for byte pair embedding and hypothesis integration layer within the shared layers, which can further enhance the MTL. The relative error reduction is 19.3% for domain classification and 36.9% for intent classification. We also use some utterances to analyze the real cause of the improvements. By decoding the hidden representations and comparing with transcription, we find by the MTL, the model attempts to find a balance point and do some reasonable query rewriting. In the future, we will explore more by introducing more tasks, improving the efficiency and utilizing more abundant information like word lattice.

References

Jon Barker, Shinji Watanabe, Emmanuel Vincent, and Jan Trmal. 2018. The fifth'chime'speech separation and recognition challenge: dataset, task and baselines. *arXiv preprint arXiv:1803.10609*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Ho Yin Chan and Phil Woodland. 2004. Improving broadcast news transcription by lightly supervised discriminative training. In *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–737. IEEE.

Michael Collins, Brian Roark, and Murat Saraclar. 2005. Discriminative syntactic language modeling for speech recognition. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 507–514. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jonathan G Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover). In *1997 IEEE Workshop on Automatic Speech Recognition and Understanding Proceedings*, pages 347–354. IEEE.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Zhiting Hu, Poyao Huang, Yuntian Deng, Yingkai Gao, and Eric Xing. 2015. Entity hierarchy embedding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1292–1300.

Kshitiz Kumar, Chaojun Liu, and Yifan Gong. 2014. Normalization of asr confidence classifier scores via confidence mapping. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Kshitiz Kumar, Ziad Al Bawab, Yong Zhao, Chaojun Liu, Benoît Dumoulin, and Yifan Gong. 2015. Confidence-features and confidence-scores for asr applications in arbitration and dnn speaker adaptation. In *INTERSPEECH*.

Mingda Li, Cristian Lumezanu, Bo Zong, and Haifeng Chen. 2018. Deep learning ip network representations. In *Proceedings of the 2018 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, pages 33–39.

Mingda Li, Weitong Ruan, Xinyue Liu, Luca Soldaini, Wael Hamza, and Chengwei Su. 2020. Improving spoken language understanding by exploiting asr n-best hypotheses. *arXiv preprint arXiv:2001.05284*.

Mingda Li. 2020. *Efficient Latent Semantic Extraction from Cross Domain Data with Declarative Language*. Ph.D. thesis, UCLA.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. *arXiv preprint arXiv:1901.11504*.

Paul Mather and Brandt Tso. 2016. *Classification methods for remotely sensed data*. CRC press.

Fabrizio Morbini, Kartik Audhkhasi, Ron Artstein, Maarten Van Segbroeck, Kenji Sagae, Panayiotis Georgiou, David R Traum, and Shri Narayanan. 2012. A reranking approach for recognition and classification of speech input in conversational dialogue systems. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 49–54. IEEE.

Atsunori Ogawa, Marc Delcroix, Shigeki Karita, and Tomohiro Nakatani. 2018. Rescoring n-best speech recognition list based on one-on-one hypothesis comparison using encoder-classifier model. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6099–6103. IEEE.

Atsunori Ogawa, Marc Delcroix, Shigeki Karita, and Tomohiro Nakatani. 2019. Improved Deep Duel Model for Rescoring N-Best Speech Recognition List Using Backward LSTMLM and Ensemble Encoders. In *Proc. Interspeech 2019*, pages 3900–3904.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Fuchun Peng, Scott Roy, Ben Shahshahani, and Françoise Beaufays. 2013. Search results based n-best hypothesis rescoring with maximum entropy classification. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 422–427. IEEE.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Jinfeng Rao, Ferhan Ture, Hua He, Oliver Jojic, and Jimmy Lin. 2017. Talking to your tv: Context-aware voice search with hierarchical recurrent neural networks. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 557–566.

Hasim Sak, Murat Saraclar, and Tunga Gungor. 2011. Discriminative reranking of ASR hypotheses with morpholexical and n-best-list features. In *2011 IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU 2011, Waikoloa, HI, USA, December 11-15, 2011*, pages 202–207.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pages 242–264. IGI Global.

Gokhan Tur and Renato De Mori. 2011. *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Wayne Xiong, Lingfeng Wu, Fil Alleva, Jasha Droppo, Xuedong Huang, and Andreas Stolcke. 2018. The microsoft 2017 conversational speech recognition system. In *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5934–5938. IEEE.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

Yu Zhang and Qiang Yang. 2017. A survey on multi-task learning.