Multi-Task Learning for parsing the Alexa Meaning Representation Language

Vittorio Perera [*] **Tagyoung Chung** [†] **Thomas Kollar** [†] **Emma Strubell** [‡]

Abstract

The Alexa Meaning Representation Language (AMRL) is a compositional graph-based semantic representation that includes fine-grained types, properties, actions, and roles and can represent a wide variety of spoken language. AMRL increases the ability of virtual assistants to represent more complex requests, including logical and conditional statements as well as ones with nested clauses. Due to this representational capacity, the acquisition of large scale data resources is challenging, which limits the accuracy of resulting models. This paper has two primary contributions. The first contribution is a linearization of the AMRL parses that aligns it to a related task of spoken language understanding (SLU) and a deep neural network architecture that uses multi-task learning to predict AMRL fine-grained types, properties and intents. The second contribution is a deep neural network architecture that leverages embeddings from the large-scale data resources that are available for SLU. When combined, these contributions enable the training of accurate models of AMRL parsing, even in the presence of data sparsity. The proposed models, which use the linearized AMRL parse, multi-task learning, residual connections and embeddings from SLU, decrease the error rates in the prediction of the full AMRL parse by 3.56% absolute.

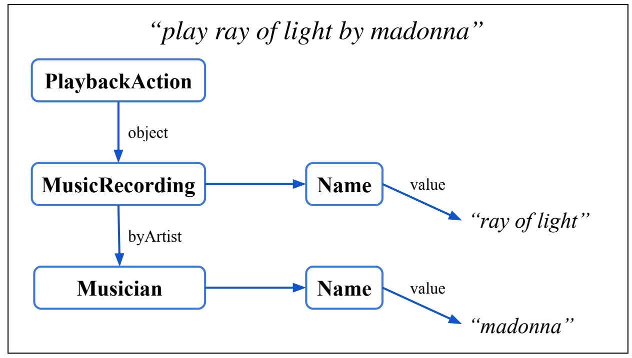


Figure 1: Meaning representation of a sentence using AMRL.

"play ray of light by madonna"

Domain: MusicApp
Intent: ListenMediaIntent
Slots: play [ray of light]$_{\text{Song}}$ by [madonna]$_{\text{Singer}}$

Figure 2: Example of the spoken language understanding (SLU) representation.

Introduction

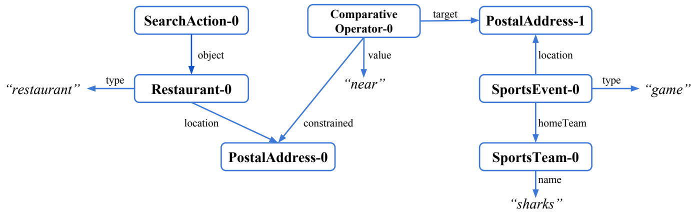
As intelligent assistants become more open and connected, there is a need to expand their capacity to understand task-oriented language. Expanding this capacity requires a representation that can handle spoken language and models that can predict this representation. Previous representations for spoken language understanding (SLU) use a fixed, flat structure, categorizing requests into domains, intents and slots (Figure 2). A domain is a general category for a request (e.g., music, calendar), an intent is an action within that domain (e.g., play, search) and slots are mentions within the utterance (e.g., "ray of light" is the song to be played). The limited expressivity of the SLU representation leads to a number of challenges. First, requests with a similar surface form may belong to different domains, which makes it challenging to add new features without degrading the accuracy of existing domains. Second, utterances that span multiple domains are not easily supported. For example, "find me a restaurant near the sharks game" would belong to both the SLU local search and the sports domains. Third, complex requests are not supported using the fixed and flat structure of SLU parses. For example, "play hunger games and turn the lights down to 3" involves multiple domains and sequential actions.

This paper uses a new representation designed for spoken language, called the Alexa Meaning Representation Language (AMRL). AMRL is a compositional, graph-based semantic representation that is backed by a large-scale ontology. An AMRL parse contains actions, fine-grained types, properties, and verb roles (as in Figure 1) and provides the representational capacity to not only understand simple requests to a virtual assistant, but also complex logical, conditional,

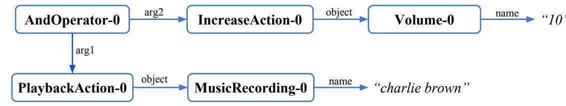
[*] Vittorio Perera (vdperera@cs.cmu.edu) is at Carnegie Mellon University.

[†] Tagyoung Chung (tagyoung@amazon.com) and Thomas Kollar (kollart@amazon.com) are at Amazon Inc.

[‡] Emma Strubell (strubell@cs.umass.edu) is at the University of Massachusetts Amherst.



(a) Cross-domain



(b) Sequential (multiple actions)

Figure 3: (a) is the AMRL for *"find restaurants near the sharks game."*. (b) is the AMRL for *"play charlie brown and turn the volume up to 10"*. There are two actions that AMRL could handle with a sequential operator. These requests are challenging to handle using the current domain-based SLU representation.

"play ray of light by madonna'	
Properties:	play [ray of light]$_{\text{object.name}}$ by [madonna]$_{\text{object.byArtist.name}}$
Entities:	play [ray of light]$_{\text{name@MusicRecording}}$ by [madonna]$_{\text{name@Musician}}$
Intents:	PlaybackAction<object@MusicRecording>

Figure 4: Linearized annotation for AMRL.

and nested statements as well as cross-domain utterances (Figure 3).

The primary contribution of this paper is a set of models that can predict AMRL parses given a natural language utterance. These models predict linearized AMRL parses that factor into three components: intents, types, and properties (Figure 4). The tasks are ordered from coarse to fine, enabling subsequent layers to take as input embeddings from previous layers. Multi-task learning is used to jointly train the parameters of the model, the simplest of which consists of a word-embedding layer, three bi-directional LSTM layers, residual connections, and a custom decoder (He et al. 2017). Large-scale resources available for SLU (e.g., 3m SLU examples versus 300k AMRL examples) provide embeddings for related tasks that improve the overall accuracy of the model. The proposed model learns to SLU domains and slots jointly with the AMRL intents, types and properties.

The use of an SLU slot embedding layer is found to produce a 2.6% improvement in IRER (full-parse accuracy), while domain embeddings improve the accuracy by another 0.1% IRER. An additional 1% IRER improvement is obtained by using a custom decoder that enforces span-based IOB and ontological constraints. Our proposed model decreases the full-parse error rate by 1.5% IRER when compared to a strong baseline model that has access to additional inputs, such as gazetteers and general-purpose word embeddings. The rest of this document is organized as follows, we review the related work, introduce the AMRL and SLU representations, the baseline and proposed models, a custom decoder and results.

Related Work

The Alexa Ontology is a version of schema.org (Guha, Brickley, and Macbeth 2016) that has been adapted for spoken language understanding and is used as the basis for the Alexa Meaning Representation Language. Some alternative semantic representations include FrameNet (Baker, Fillmore, and Lowe 1998), lambda-DCS (Liang 2013), combinatory categorial grammars (CCG) (Steedman and Baldridge 2011) and universal dependencies (Nivre and others 2016). AMR is a hierarchical graph-based representation targeted for longer texts (Banarescu and others 2013; Kevin Knight 2017). Unlike these approaches, AMRL focuses on directly supporting spoken language understanding and contains fine-grained types along with actions, verb roles, and properties. An overview of other related semantic representations is covered in Abend and Rappoport.

DNNs are widely used for sequence labeling. Shimaoka et al. perform fine-grained entity labeling using a neural attention model. Dong et al. use a combination of NNs to embed words and entities for coarse-grained entity labeling. More recently, two types of network architectures have gained popularity. The first one is LSTM-CNNs (Chiu and Nichols 2015), which use a combination of word-level and CNN-extracted character-level features to augment the input to bi-LSTMs. The second one is LSTM-CRFs (Huang, Xu, and Yu 2015), which apply a CRF constraints to bi-LSTMs. Recently, Ma and Hovy combined the two approaches to get the state of the art results on standard CONLL 2003 NER task.

LSTMs (Long Short Term Memory) (Hochreiter and Schmidhuber 1997) perform well on many NLP tasks including sequence tagging, intent classification, and language modeling due to their inherent ability to model long term sequential dependencies. Bi-LSTMs (Graves, rahman Mohamed, and Hinton 2013) are layered architectures which effectively use past and future information via Forward and Backward LSTM layers. Bi-LSTMs have been successfully applied to feature generation for tasks like dependency parsing (Kiperwasser and Goldberg 2016) and semantic role labeling (Zhou and Xu 2015). All our models adopt a deep neural-network architecture with bi-LSTMs as our primary building blocks. For text classification, there has been a lot of recent interest in using character-level embeddings (Kim et al. 2015) as an additional input to neural architectures because of their ability to model morphological features as well as effectively handle out-of-vocabulary words. Ballesteros, Dyer, and Smith use character embedding for dependency parsing, Xiao and Cho combine character embeddings with CNNs for the text classification task.

Semantic parsing and spoken language understanding (SLU) learn to map natural language to a formal representation. Although semantic parsers can be trained using sentences annotated with this formal representation (Zelle and Mooney 1996; Zettlemoyer and Collins 2012; Wong and Mooney 2006; Kwiatkowski et al. 2010; Krishnamurthy and Mitchell 2012), they have not generally been applied to spoken language. Most applications of spoken language understanding map utterances onto a fixed domain, intent, and slot structure (Gupta et al. 2006), which cannot represent complex, cross-domain, or compositional utterances.

Multitask learning in deep neural networks has been shown to help generalization. Similar to our work but with CNNs, Xu and Sarikaya jointly model sentence classification and sequence labeling. Guo et al. use recursive neural networks to jointly classify intents and fill slots. Miwa and Bansal achieved the state of the art for entity and relation classification. Zhang and Weiss used them for part of speech tagging and dependency parsing. Transferring of learned embeddings was explored in (Yosinski et al. 2014). Our work builds on these by creating a deep multi-task model for predicting AMRL.

Approach

In this section we describe the Alexa Meaning Representation Language (AMRL), the proposed baseline neural network architecture, the multi-task learning approach used for training, the use of learned embeddings from the related SLU prediction tasks and a custom decoder for constrained inference.

Alexa Meaning Representation Language

AMRL provides a common semantics for spoken language. An AMRL parse consists of five primary components:

- **Actions** define the core functionality used in spoken language. In Figure 1, the PlaybackAction is used on a MusicRecording, but can also be used on Books, Videos and other playable objects.
- **Roles** Actions operate on entities via roles. The .object role defines the entity on which the action operates.
- **Types** apply to each entity mention in a request and are hierarchical. In Figure 1, there is a MusicRecording and Musician type.
- **Properties** define relationships between instances of types. For example, the "byArtist" property of a MusicRecording, defines who wrote a song.
- **Operators** Operators can be used to represent complex logical or spatial relationships.

The aim of AMRL is to provide a common semantic representation for spoken language. It can represent anaphora, conditional statements, sequential actions, and logical expressions. AMRL can have arbitrary nesting, enabling it to represent complex statements. Figure 1 shows a simple example of AMRL. Figure 3 shows how sequential and cross-domain queries can be represented in AMRL.

Data sparsity is addressed by linearizing AMRL parses such that they align with the SLU tasks of slot prediction and intent prediction. An example of the linearization scheme for AMRL can be seen in Figure 4. The linearization uses the full AMRL representation, which is a rooted graph. Starting at the root, the property linearization recursively descends nodes in the graph, appending each property visited along the way until a leaf node is reached. Incoming property arcs to a visited node are inverted to avoid cycles and handle multi-headed graphs. Types of the leaf nodes are the only ones that appear in the linearization. When a leaf node is visited, the corresponding leaf property is included. Leaf properties include "name", which is the mention of an entity (e.g., "McDonalds"), "type", which is the mention of the type of an entity (e.g., Restaurant) and value, which is the mention of a numeric value. Intents include action, the roles on the action (e.g., "object"), and the type of each of the roles. This linearization enables the AMRL to be factored into three components: intents, types, and properties (Figure 4).

Models

The proposed model for AMRL parsing is a deep neural network architecture that uses multi-task learning. In this section we describe the baseline model used in our experiments and extensions to the baseline model that leverage embedding layers from SLU. Multi-task learning exploits commonalities between the SLU and AMRL tasks (Caruana 1998), such as overlapping spans.

Baseline model The baseline model is a deep bi-directional LSTM neural network trained using multi-task learning. The architecture is structured as a coarse-to-fine prediction problem. There are three LSTM layers, each of which predicts a different component of the parse. The first layer performs coarse-grained type prediction, the next layer performs fine-grained property prediction and the final layer performs intent classification. Residual connections are included to leverage embeddings from previous stages. For example, the word embeddings are used as input to the LSTM layers predicting the properties and the actions.

Figure 5 shows the topology of this model, where each block is a bi-directional LSTM. The first layer provides a shared word embedding (Collobert and Weston 2008), while the remaining three LSTM layers are connected to an affine transform and a softmax layer to provide the output for each of the three tasks (i.e., properties, type, and intent prediction). At the output layers, we form a prediction by feeding the concatenated hidden representations for the token we wish to label into another affine transform followed by a softmax to obtain posterior probabilities over the set of labels. Dropout is applied after each LSTM layer. For classification, only the final state from the forward and backward LSTM is used to predict the category. IOB tagging is used to denote the inside, outside, and beginning of each property and type span. The input of this baseline model is a one-hot vector for each word in the sentence.

Transferring learned representations Four neural network architectures have been investigated as a means to

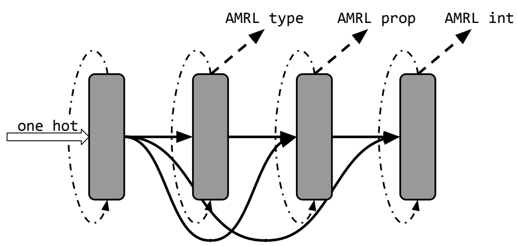


Figure 5: Topology of the baseline multi-task DNN model. Each component (above) is a bi-LSTM layer. The final layer is a softmax, and the dashed line is the recurrent connection. "type" and "prop" perform sequential prediction, while "int" categorizes the intent of the user.

include SLU domain and slot embedding layers into the baseline AMRL parsing model. Intent embeddings were added to initial models, but did not result significant performance gains.

The first two architectures leverage embeddings from SLU slots. Since SLU slots and AMRL types are often correlated in terms of the spans, we experiment with two mechanisms for incorporating embeddings. The first architecture is to use a common set of weights trained on both SLU slot and AMRL type tasks. The second architecture is to train an SLU slot embedding and use this embedding as an input to AMRL type prediction. To investigate the former, a model (+SLU Slots) is developed to predict SLU slots in parallel to the AMRL types. The resulting model can be seen in Figure 6a. To investigate the latter, a model (+SLU Slots pipe) is developed to predict the SLU slots before predicting the AMRL types. The embedding from the SLU slot layer is used to to predict the later stages of AMRL. This model can be seen in Figure 6b.

The remaining two modeling architectures leverage both domain and slot embeddings. Since the domain is a strong indicator of the entities and their relationships, we anticipate that it could be incorporated in a similar way. The first architecture trains a common set of weights for predicting SLU slots and domains, and uses the resulting embedding as input to later stages. The second trains the domain embedding as a parallel task that can be used to as input to later layers that predict AMRL types and properties. An embedding for SLU slots was used since it was was found to be more accurate in our initial experiments.

In the first model (+SLU Slots/Domain), the domain layer is trained as a task in parallel to the AMRL type layer and is input to the property layer. The rationale behind this choice is that, although domain and property prediction are very different tasks, the domain, when correctly classified, can restrict what kind of properties we expect (e.g., if a sentence is classified in the *Music* domain we expect properties such as *byArtist* but not *weatherCondition*). There is still a shared embedding space learned as input to the type model. This model (+SLU Slots/Domain) is shown in Figure 6c.

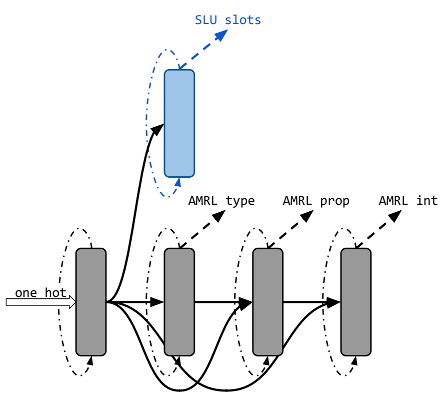
In the second model (+SLU Slots/Domain pipe), a common LSTM embedding layer is trained for SLU slots and domains. This layer is input to the layer that predicts the AMRL types. The hypothesis is that a single LSTM embedding layer might be enough to encode all the information from the SLU representation. Figure 6d shows the topology for this network.

Word embeddings and gazetteers Pre-trained word embeddings and gazetteers are added as an input to our baseline model. Three hundred dimensional pre-trained word2vec embeddings are used, trained on the Google News corpus on 100 billion utterances (Mikolov et al. 2013) and are incorporated as an additional input to the one-hot encoding of each word. Gazetteers (lists of entity mentions) from the Alexa ontology that backs AMRL were used in a similar way to word embeddings (e.g., as an additional input per word). For example, a Musician gazetteer will contain a list of music of musician names like "Sting." These features are indicators that are set to 1 if the current word or word sequence appears in a gazetteer, and set to 0 otherwise. This model uses the same topology of the baseline, shown in Figure 5.

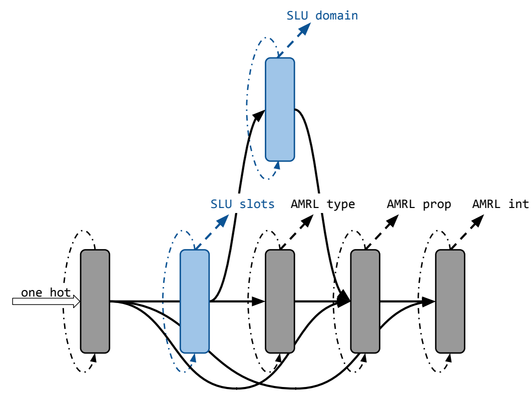
Decoding A beam-search decoder was written to leverage two primary constraints of this model in order to improve the accuracy of the result. The first constraint is to ensure the sequence of labels obey the IOB constraints. These constraints include (1) that an $I-$ tag follows either another $I-$ tag or a $B-$ tag with the same label, and (2) a $B-$ tag follows only an $O-$ or an $I-$ tag. The second constraint ensures that the final property label (e.g., name, type, or value) matches the initial property label, which ensures that we produce predictions consistent with the AMRL ontology. For example, in Figure 4, if "ray of light" were predicted as "type@MusicRecording" but as a property was predicted at "object.name," then this would be an invalid transition. The beam search limits property candidates to those that are above minimum probability. The value of the lower bound probability determines how aggressive the pruning is. The search is performed by combining all the property candidates with all the possible matching types.

Optimization Stochastic gradient descent with a fixed learning rate is used to optimize the parameters of the model. Both fine-tuning of learned parameters and joint training (e.g., learning the SLU embedding layers at the same time as the AMRL ones) are used. Each model is trained until it fully converges on the training set, which typically takes around 60 epochs. We use a fixed learning rate of 0.0005 with an L2 penalty of 1e-8, and a batch size of 128 sentences each. The output of each bi-LSTM layer is a vector of size 256 that is created by concatenating the hidden representation of forward and backward LSTMs (each of size 128).

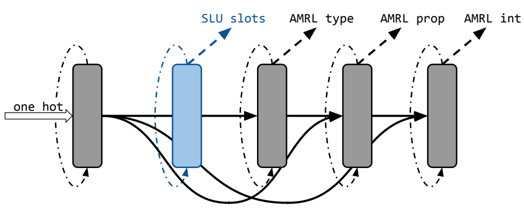
To prevent overfitting to our training set we take two precautions. First, the output of each bi-LSTM layer is connected to a drop-out component with retention rate of 80%. The output of the drop-out component is then used as input for the following layers. Second, a weighted cross-entropy loss function with a weight of λ is used in the joint training setup. Accuracy on the development set is used to evaluate generalization performance.



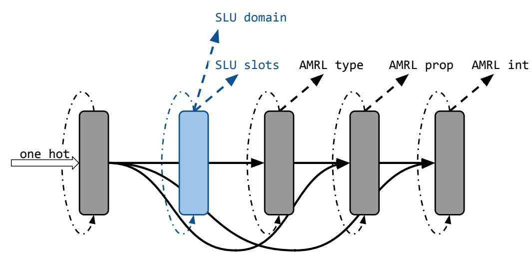
(a) Topology of the +SLU Slots model.



(c) Topology of the +SLU Slots/Domain model



(b) Topology of the +SLU Slots Pipe model



(d) Topology of the +SLU Slots/Domain Pipe model

Figure 6: Multi-task models used for training AMRL models alongside slot and domain embeddings from the existing SLU system. Highlighted in blue the bi-LSTM layers added compared to the baseline model.

Results

Datasets

The two datasets used in these experiments are (1) a large corpus collected for spoken language understanding (SLU) and (2) a smaller corpus of linearized AMRL parses. The SLU corpus is composed of a total of ∼2.8m unique sentences. These sentences span on the order of 20 domains, 200 intents, and 200 slots. The vocabulary of this dataset amounts to ∼150k distinct words. The representation for this corpus is shown in Figure 2. The AMRL corpus is significantly smaller then the SLU corpus and contains only around 350k unique sentences in the linearized representation. This data consists of intents, types and properties. Figure 4 shows an example of the annotation for the same sentence shown in Figure 1. The AMRL corpus spans over 60 linearized intents, 200 properties and 200 types. The total vocabulary of this corpus is around 42k words, one order of magnitude smaller then the SLU one. The development set and test set contain around 48k sentences annotated using AMRL. Accuracy is reported only on the AMRL test set.

Metrics

Four metrics are considered to evaluate our models. The first two, $F1_{IC}$, $F1_{SC}$, are F1 scores evaluated respectively at intent and slot level. $F1_{SC}$ is a strong metric, as it requires the spans and labels for both the property and the type tasks to be correct.

The Intent Classification Error Rate (ICER) is inversely correlated with the $F1_{IC}$ and measures the number of incorrect intents. ICER is not always sufficient as there may be multiple intents in an utterance. The formula for ICER is:

$$\frac{\#\textit{incorrect(intents)}}{\#\textit{total intents}}$$

Finally, the Intent Recognition Error rate (IRER) is computed as:

$$\frac{\#\textit{incorrect(interpretation)}}{\#\textit{total utterances}}$$

where we consider an interpretation incorrect if any of the slots or intents differs from the ground truth.

Evaluation

Table 1 shows the results for different model architectures. The baseline model is trained using only the AMRL corpus. The vocabulary is pruned and only words appearing twice or more are used. Every other word is mapped to an "unknown" token resulting in a vocabulary of ∼25k words. From these results it appears that using embeddings from SLU tasks provides benefit; our best model (+SLU Slots/Domain) outperforms the baseline across all the metrics and some by a considerable margin.

$Models$	$F1_{IC}$	$F1_{SC}$	$ICER$	$IRER$
Baseline	0.9383	0.8439	6.4464	25.7176
+SLU Slots	0.9416	0.8551	6.1254	24.2876
+SLU Slots (jt)	0.9389	0.8449	6.4587	25.6867
+SLU Slots pipe	0.9432	0.8602	**5.8867**	23.1312
+SLU Slots pipe (jt)	0.9390	0.8456	6.3311	25.3534
+SLU Slots/Domain	0.9431	0.8614	5.9649	**23.0222**
+SLU Slots/Domain (jt)	**0.9435**	**0.8621**	6.0204	23.1147
+SLU Slots/Domain pipe	0.9351	0.8232	6.8270	29.9418
+SLU Slots/Domain pipe (jt)	0.9400	0.8538	6.1789	24.4501

Table 1: Models and their results. Models marked as (jt) are trained using the joint training approach.

We also compared fine-tuning pretrained embeddings and joint training (learning the SLU embedding layers at the same time as the other embeddings). For pretraining, a network is trained to predict the SLU tasks; once converged the AMRL-specific components (i.e., the last three LSTM layers of each model) are added and trained until convergence using a cross-entropy loss. Joint training optimizes both SLU tasks and AMRL ones at the same time. At each time-step a random training instance is selected from one of the two corpora with probability ($p = |AMRL|/|SLU|$). Since the size of the SLU corpus is much bigger then the AMRL one, we must prevent overfitting on the SLU tasks. To do so a weighted cross entropy loss function is used, where the SLU tasks are down-weighted by a factor $\lambda = |AMRL|/(10 \times |SLU|)$.

Joint training results in a slight improvement in accuracy of the proposed model, though the results are inconclusive. For the first two models pretraining appears to result in higher accuracy but for the remaining two the opposite seems to hold true. One possible explanation for this behavior is that the cross entropy loss function is used in conjunction with the joint training approach. In our experiments we fixed the weight for the loss function, but additional hyper-parameter tuning might improve the overall result.

The accuracy of baseline model was also compared against the best performing model across different actions. Figure 7 shows that our dataset is skewed, with two of the actions (Playback and Search) covering more then 97% of the total training instances; the remaining 15 actions have an almost uniform amount of sentences. Table 2 shows the gain in ICER for each action. We observe that the improvement is modest on the two most represented actions but much more pronounced on the less represented ones. In general, we find that when there is sufficient data available (i.e., for the Playback and Search actions) adding more information from the SLU task is not helpful. On the other hand, when fewer training instances are available the information provided by the SLU tasks becomes more valuable and strongly improves the results.

In Table 3, the best models are compared against the baseline model which has access to general-purpose word embeddings and gazetteers. The baseline model, which only has access only to the AMRL training dataset has lowest accuracy. Adding gazetteer and general-purpose word embeddings improves accuracy, though the best model is the one that transfers the learned representations from the SLU task (proposed model from Table 1). Incorporating the additional constraints in the decoder (e.g., IOB and final property) results in our best model. For these experiments, we used a decoder with a beam of size 3 and a minimum probability of 10^{-7}. As expected the decoder does not impact any of the intent metrics ($F1_{IC}$ and ICER). The structural incorrectness of the predicted outputs is upper bounded by 0.96% IRER using our proposed model with the custom decoder.

Conclusion

AMRL is a new graph-based representation for the meaning of a sentence. Since annotating AMRL is time consuming and costly, only a limited amount of data is available. In this paper we show that learned embeddings from related tasks can improve the accuracy of AMRL models. Domain and slot embeddings help significantly, improving the accuracy by 3.56% IRER (full-parse accuracy). A constrained decoder that leverages IOB and type/property constraints is a key component, decreasing IRER by 1% absolute.

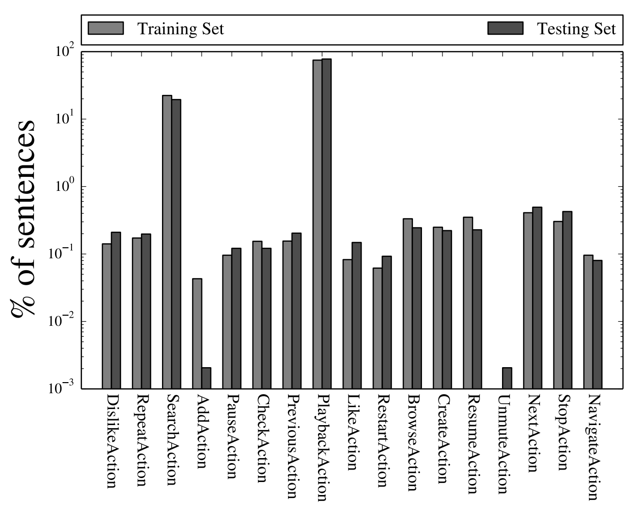


Figure 7: Action distribution on the training and testing set for the AMRL corpus.

Actions	*Training instances*	Δ *ICER*
PlaybackAction	HIGH	-0.53
SearchAction	MED	-0.17
NextAction	LOW	-3.34
ResumeAction	LOW	-10.81
BrowseAction	LOW	+5.88
StopAction	LOW	-1.93
CreateAction	LOW	0
RepeatAction	VLOW	+3.13
PreviousAction	VLOW	-23.23
CheckAction	VLOW	-5.08
DislikeAction	VLOW	-1.96
PauseAction	VLOW	-11.86
NavigateAction	VLOW	-5.12
LikeAction	VLOW	-6.94
RestartAction	VLOW	0

Table 2: Δ at ICER. The HIGH bin contains more than 100k examples. The MED bin contains between 2k and 100k examples. The LOW bin contains between 800 and 2k examples. The VLOW bin contains between 100 and 800 examples. The actions are ordered, in decreasing fashion, based on the number of occurrences in the AMRL training set.

	$F1_{IC}$	$F1_{SC}$	$IRER$
baseline	0.93	0.84	25.71
baseline+emd+gaz	0.94	0.86	23.65
proposed model	0.94	0.86	23.11
proposed model+decoder	0.94	**0.87**	**22.15**

Table 3: Results as compared to various baselines. Baseline is the multi-task model trained only on AMRL data. Baseline+emb+gaz is the baseline model with word embedding and gazetteer features as input to the model. Proposed model is the best model without word embeddings or gazetteer features. Proposed model+decoder includes results after decoding of the best model (without gazetteers or word embeddings as input). Beam size is three and a floor probability of 10^{-7} is used.

References

Abend, O., and Rappoport, A. 2017. The state of the art in semantic representation. In *Proc. of ACL*.

Baker, C. F.; Fillmore, C. J.; and Lowe, J. B. 1998. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98, 86–90. Stroudsburg, PA, USA: Association for Computational Linguistics.

Ballesteros, M.; Dyer, C.; and Smith, N. A. 2015. Improved transition-based parsing by modeling characters instead of words with lstms. *arXiv preprint arXiv:1508.00657*.

Banarescu, L., et al. 2013. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, 178–186. Sofia, Bulgaria: Association for Computational Linguistics.

Caruana, R. 1998. Multitask learning. In *Learning to learn*. Springer. 95–133.

Chiu, J. P., and Nichols, E. 2015. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*.

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

Dong, L.; Wei, F.; Sun, H.; Zhou, M.; and Xu, K. 2015. A hybrid neural model for type classification of entity mentions. In *Proceedings of the 24th International Conference on Artificial Intelligence*, 1243–1249. AAAI Press.

Graves, A.; rahman Mohamed, A.; and Hinton, G. 2013. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Guha, R. V.; Brickley, D.; and Macbeth, S. 2016. Schema.org: Evolution of structured data on the web. *Commun. ACM* 59(2):44–51.

Guo, D.; Tur, G.; Yih, W.-t.; and Zweig, G. 2014. Joint semantic utterance classification and slot filling with recursive neural networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, 554–559. IEEE.

Gupta, N.; Tur, G.; Hakkani-Tur, D.; Bangalore, S.; Riccardi, G.; and Gilbert, M. 2006. The at&t spoken language understanding system. *IEEE Transactions on Audio, Speech, and Language Processing* 14(1):213–222.

He, L.; Lee, K.; Lewis, M.; and Zettlemoyer, L. 2017. Deep semantic role labeling: What works and whats next. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional lstm-crf models for sequence tagging. In *arXiv preprint arXiv:1508.01991*.

Kevin Knight, Bianca Badarau, L. B. C. B. M. B. K. G. U. H. D. M. M. P. T. O. N. S. 2017. Abstract meaning representation (amr) annotation release 2.0.

Kim, Y.; Jernite, Y.; Sontag, D.; and Rush, A. M. 2015. Character-aware neural language models. *arXiv preprint arXiv:1508.06615*.

Kiperwasser, E., and Goldberg, Y. 2016. Simple and accurate dependency parsing using bidirectional lstm feature representations. *Transactions of the Association for Computational Linguistics (TACL)* (4):313327.

Krishnamurthy, J., and Mitchell, T. M. 2012. Weakly supervised training of semantic parsers. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 754–765. Association for Computational Linguistics.

Kwiatkowski, T.; Zettlemoyer, L.; Goldwater, S.; and Steedman, M. 2010. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, 1223–1233. Stroudsburg, PA, USA: Association for Computational Linguistics.

Liang, P. 2013. Lambda dependency-based compositional semantics. *CoRR* abs/1309.4408.

Ma, X., and Hovy, E. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

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

Miwa, M., and Bansal, M. 2016. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770*.

Nivre, J., et al. 2016. Universal dependencies v1: A multilingual treebank collection. In Chair), N. C. C.; Choukri, K.; Declerck, T.; Goggi, S.; Grobelnik, M.; Maegaard, B.; Mariani, J.; Mazo, H.; Moreno, A.; Odijk, J.; and Piperidis, S., eds., *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Paris, France: European Language Resources Association (ELRA).

Shimaoka, S.; Stenetorp, P.; Inui, K.; and Riedel, S. 2016. Neural architectures for fine-grained entity type classification. *arXiv preprint arXiv:1606.01341*.

Steedman, M., and Baldridge, J. 2011. *Combinatory Categorial Grammar*. Wiley-Blackwell. 181–224.

Wong, Y. W., and Mooney, R. J. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 439–446. Association for Computational Linguistics.

Xiao, Y., and Cho, K. 2016. Efficient character-level document classification by combining convolution and recurrent layers. *arXiv preprint arXiv:1602.00367*.

Xu, P., and Sarikaya, R. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, 78–83. IEEE.

Yosinski, J.; Clune, J.; Bengio, Y.; and Lipson, H. 2014. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, 3320–3328.

Zelle, J. M., and Mooney, R. J. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, 1050–1055.

Zettlemoyer, L. S., and Collins, M. 2012. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*.

Zhang, Y., and Weiss, D. 2016. Stack-propagation: Improved representation learning for syntax. *arXiv preprint arXiv:1603.06598*.

Zhou, J., and Xu, W. 2015. End-to-end learning of semantic role labeling using recurrent neural networks. In *Transactions of the Association for Computational Linguistics (TACL)*.