Simple, Fast, Accurate Intent Classification and Slot Labeling for Goal-Oriented Dialogue Systems

Arshit Gupta[*]
Amazon AI
Seattle
arshig@amazon.com

John Hewitt[*] [†]
Stanford University
Palo Alto
johnhew@stanford.edu

Katrin Kirchhoff
Amazon AI
Seattle
katrinki@amazon.com

Abstract

With the advent of conversational assistants like Amazon Alexa, Google Now, etc., dialogue systems are gaining a lot of traction, especially in industrial settings. These systems typically include a Spoken Language understanding component which consists of two tasks: Intent Classification (IC) and Slot Labeling (SL). Generally, these two tasks are modeled together jointly to achieve best performance. However, this joint modeling adds to model obfuscation. In this work, we first design framework for a modularization of joint IC-SL task to enhance architecture transparency. Then, we explore a number of self-attention, convolutional, and recurrent models, contributing a large-scale analysis of modeling paradigms for IC+SL across two datasets. Finally, using this framework, we propose a class of 'label-recurrent' models that are non-recurrent apart from a 10-dimensional representation of the label history, and show that our proposed systems are highly accurate (achieving over 30% error reduction in SL over the state-of-the-art on the Snips dataset), as well as fast, at 2x the inference and 2/3 to 1/2 the training time of comparable recurrent models, thus giving an edge in critical real-world systems.

1 Introduction

At the core of task-oriented dialogue systems are spoken language understanding (SLU) models, tasked with determining the intent of users' utterances and labeling semantically relevant words at each turn of the conversation. Performance on these tasks, known as intent classification (IC) and slot labeling (SL), upper-bounds the utility of such dialogue systems. A large body of recent research has improved these models through the use of recurrent neural networks, encoder-decoder architectures, and attention mechanisms. However, for production dialogue systems in particular, system speed is at a premium, both during training and in real-time inference.

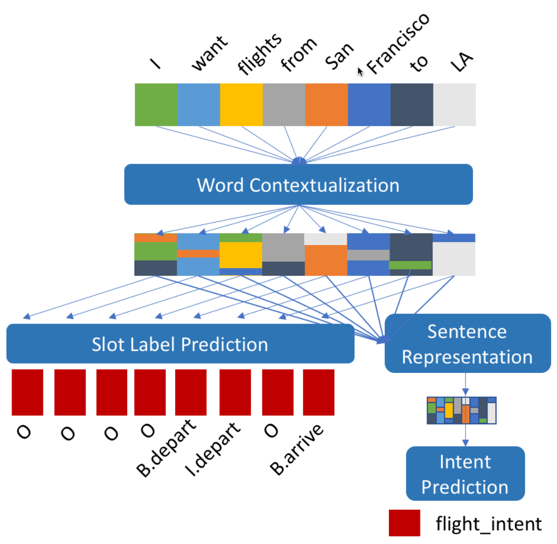

Figure 1: A general framework of joint IC+SL, decoupling modeling tasks to permit the analysis of each component independently.

In this work, we propose fully non-recurrent and label-recurrent model paradigms including self-attention and convolution for comparison to state-of-the-art recurrent models in terms of accuracy and speed. To achieve this, we design a framework for joint IC-SL models that is modularized into different components and makes the task agnostic to type of neural network used. This, in turn, makes the model architecture simpler, easy to understand and renders the task network agnostic, allowing for easier plug and play using existing components, such as pre-trained contextual word embeddings (Devlin et al., 2019), etc. This is essential for easier model debugging and quicker experimentation, especially in industrial settings.

Using this framework, we identify three distinct model families of interest: fully recurrent,

[*]Equal Contribution
[†]Work performed while at Amazon AI

label-recurrent, and non-recurrent. Recent state-of-the-art models fall into the first category, as encoder-decoder architectures have recurrent encoders to perform word context encoding, and predict slot label sequences using recurrent decoders that use both word and label information as they decode (Hakkani-Tür et al., 2016; Liu and Lane, 2016; Li et al., 2018). In second category, we have 'non-recurrent' networks: fully feed-forward, attention-based, or convolutional models, for example. Lastly, we have a class of label-recurrent models, inspired by structured sequential models like conditional random fields on top of non-recurrent word contextualization components. In this class of models, slot label decoding proceeds such that label sequences are encoded by a recurrent component, but word sequences are not.

Our contributions are:

- A class of label-recurrent convolutional models that achieve state-of-the-art performance on Snips and competitive performance on ATIS while maintaining faster training and inference speeds than fully-recurrent models
- A new modular framework for joint IC-SL models that permits the analysis of individual modeling components that decomposes these joint models into separate components for *word context encoding*, *summarization of the sentence* into a single vector for intent classification, and *modeling of dependencies in the output space* of slot label sequences.
- In-depth analysis of different word contextualizations for Spoken Language Understanding task (for instance, providing evidence for the intuition that explicitly focusing on local context is a useful architectural inductive prior for slot labeling)

2 Prior Work

There is a large body of research in applying recurrent modeling advances to intent classification and slot labeling (frequently called spoken language understanding). Traditionally, for intent classification, word n-grams were used with SVM classifier (Haffner et al., 2003) and Adaboost (Schapire and Singer, 2000). For the SL task, CRFs (Gorin et al., 1997) have been used in the past.

Recently, a larger focus has been on joint modeling of IC and SL tasks. Long short-term memory recurrent neural networks (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit models (Cho et al.) were proposed for slot labeling by Yao et al. (2014) and Zhang and Wang (2016) respectively, while Guo et al. (2014) used recursive neural networks. Subsequent improvements to recurrent neural modeling techniques, like bidirectionality and attention (Bahdanau et al., 2014) were incorporated into IC+SL in recent years as well (Hakkani-Tür et al., 2016; Liu and Lane, 2016). Li et al. (2018) introduced a self-attention based joint model where they used self-attention and LSTM layers along with the gating mechanism for this task.

Non-recurrent modeling for language has been re-visited recently, even as recurrent techniques continue to be dominant. Dilated CNNs (Yu and Koltun, 2015) with CRF label modeling were applied to named entity recognition by Strubell et al. (2017), and earlier were applied to SL by Xu and Sarikaya (2013). Convolutional and attention-based sentence encoders have been applied in complex tasks, including machine translation, natural language inference, and parsing. (Gehring et al., 2017; Vaswani et al., 2017; Shen et al., 2017; Kitaev and Klein, 2018) We draw from both of these bodies of work to propose a simple yet highly effective family of IC+SL models.

3 A general framework of joint IC+SL

Intent classification and slot labeling take as input an utterance $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_T\}$, composed of words $\mathbf{x}_i$ and of length T. Models construct a distribution over intents and slot label sequences given the utterance. One intent is assigned per utterance and one slot label is assigned per word:

$$P(l_{1:T}, c\,|\mathbf{x}_{1:T}) \qquad (1)$$

where $c \in \mathcal{I}$, a fixed set of intents, and $l_i \in \mathcal{L}$, a fixed set of slot labels. Models are trained to minimize the cross-entropy loss between the assigned distribution and the training data. To the end of constructing this distribution, our framework explicitly separates the following components, which are explicitly or implicitly present in all joint IC+SL systems (Figure 1):

3.1 Word contextualization

We first assume words are encoded through an embedding layer, providing context-independent word vectors. Overloading notation, we denote the embedded sequence $\mathbf{x}_{1:T}$, with $\mathbf{x}_i \in \mathbb{R}^{d_x}$.

In this component, word representations are enriched with sentential context. Each word $\mathbf{x}_i$ is assigned a contextualized representation $\mathbf{h}_i$. To ease layering these components, we keep the dimensionality the same as the word embeddings; $\mathbf{h}_i \in \mathbb{R}^{d_x}$. Our study consists mainly of varying this component across models, which are described in detail in Section 4. In all models, we assume independence of intent classification and slot labeling given the learned representations:

$$P(l_{1:T}, c|\mathbf{h}_{1:T}) = P(l_{1:T}|\mathbf{h}_{1:T})P(c|\mathbf{h}_{1:T}) \quad (2)$$

3.2 Sentence representation

In this component, the output of the word contextualization component is summarized in a single vector,

$$\mathbf{s} = \text{SentenceRepr}(\mathbf{h}_{1:T}) \quad (3)$$

where $\mathbf{s} \in \mathbb{R}^{d_x}$. For all our experiments, we keep this component constant, using a simple attention-like pooling which is the weighted sum of word contextualization for each position in the sentence. These weights are computed using softmax over these individual word contextualizations

While simple, this model permits word contextualization components freedom in how they encode sentential information; for example, self-attention models may spread full-sentence information across all words, whereas 1-directional LSTMs may focus full-sentence information in the last word's vector.

3.3 Intent prediction

In this component, the sentence representation is used as features to predict the intent of the utterance. For all experiments, we keep this component fixed as well, using a simple two-layer feed-forward block on top of $\mathbf{s}$.

3.4 Slot label prediction

In this component, the output of the word contextualization component is used to construct a distribution over slot label sequences for the utterance. We decompose the joint probability of the label sequence given the contextualized word representations into a left-to-right labeling:

$$P(l_{1:T}|\mathbf{h}_{1:T}) = \prod_{i=1}^{T} P(l_i|\mathbf{h}_{1:T}, l_{1:i-1}) \quad (4)$$

In our experiments, we explore two models for slot prediction, one fully-parallelizable because of strong independence assumptions, the other permitting a constrained dependence between labeling decisions that we call 'label-recurrent'.

Independent slot prediction The first is a non-recurrent model, which assumes indepdencence between all labeling decisions once given $\mathbf{h}_{1:T}$, as well as independence from all word representations except that of the word being labeled:

$$P(l_i|\mathbf{h}_{1:T}, l_{1:i-1}) = P(l_i|\mathbf{h}_i) \quad (5)$$

This model is fully parallelizable on GPU architectures, and the probability of each labeling decision is modeled according to

$$P(l_i|\mathbf{h}_{1:T}) = \text{softmax}(W^{(3)}\mathbf{p}_{i,\text{SL}} + \mathbf{b}^{(3)}) \quad (6)$$
$$\mathbf{p}_{i,\text{SL}} = \tanh(W^{(4)}\mathbf{h}_i + \mathbf{b}^{(4)}) \quad (7)$$

hence, SL prediction features are learned using each contextualized word independently.

Label-recurrent slot prediction The second class of slot prediction models we consider lead to our classification, 'label-recurrent.'[1] These models permit dependence of labeling decisions on the sequence of decisions made so far, but keep the independence assumption on the word representations:

$$P(l_i|\mathbf{h}_{1:T}, l_{1:i-1}) = P(l_i|l_{1:i-1}, \mathbf{h}_i) \quad (8)$$

Notably, this family of models excludes traditional encoder-decoder models, since the decoder component uses labeling decisions $l_{1:i-1}$ and earlier word representations $\mathbf{h}_{1:i-1}$ to influence the predictor features $\mathbf{p}_{i,SL}$. However, it includes models such as CNN-CRF.

The space of label sequences in slot labeling is much smaller than the space of word sequences. This adds minimal computational burden and permits the model to benefit from GPU parallelism during $\mathbf{h}_{1:T}$ computation.

For our experiments, we propose a single label-recurrent model, which encodes labeling histories $l_{1:-i}$ using only a 10-dimensional LSTM. First, slot labels are embedded, such that for each $l \in \mathcal{L}$, we have $\mathbf{l} \in \mathbb{R}^{d_l}$. An initial tag history state, h_0^{tag}, is randomly initialized. Each tag decision is fed

[1] We use this term for clarity in language, not to claim that no such models have been explored in the past.

along with the previous tag history state to the LSTM, which returns the next tag history state:

$$\mathbf{h}_i^{\text{tag}} = \text{LSTM}(\mathbf{l}_{i-1}, \mathbf{h}_{i-1}^{\text{tag}}). \tag{9}$$

We omit a precise description of the LSTM model here for space, referring the reader to (Hochreiter and Schmidhuber, 1997).

The tag history is used at each prediction step as additional inputs to construct the predictor features $\mathbf{p}_{i,\text{SL}}$, replacing Eqn. 7 with:

$$\mathbf{p}_{i,\text{SL}} = \tanh(W^{(5)}[\mathbf{h}_i; \mathbf{h}_i^{\text{tag}}] + \mathbf{b}^{(5)}) \tag{10}$$

where $[a; b]$ denotes concatenation. This model and other label-recurrent models are not only parallelizable more than fully-recurrent models, but also provide an architectural inductive bias, separating modeling of tag sequences from modeling of word sequences. In our experiments, we perform greedy decoding to maintain a high decoding speed.

4 Word contextualization models

In this section, we describe word contextualization models with the goal of identifying non-recurrent architectures that achieve high accuracy and faster speed than recurrent models.

4.1 Feed-forward model

In this model, we set $\mathbf{h}_{1:T} = \mathbf{x}_{1:T} + \mathbf{a}_{1:T}$, where $\mathbf{a}_{1:T}$ is a learned absolute position representation, with one vector learned per absolute position, as used in (Gehring et al., 2017). While extremely simple, this model provides a useful baseline as a totally context-free model. It also permits us to analyze the contribution of a label-recurrent component in such a context-deprived scenario.

4.2 Self-attention models

Recent work in non-recurrent modeling has surfaced a number of variants of attention-based word context modeling.

The simplest constructs each $\mathbf{h}_i$ by incorporating a weighted average of the rest of the sequence, $\mathbf{x}_{1:T} \backslash \mathbf{x}_i$. We use a bilinear attention mechanism with a residual connection while masking out the identity in the attention weights.

$$\mathbf{h}_i = \text{relu}(\sqrt{.5}(\mathbf{c}_i + \mathbf{x}_i)) \tag{11}$$

$$\mathbf{c}_i = \sum_{j=1, j\neq i}^{T} \alpha_j \mathbf{x}_j \tag{12}$$

$$\alpha_j = \frac{\exp(\mathbf{x}_i^T W^{(5)} \mathbf{x}_j)}{\sum_{j'=1}^{T} \exp(\mathbf{x}_i^T W^{(5)} \mathbf{x}_{j'})} \tag{13}$$

In this and all subsequent models, we optionally stack multiple layers, feeding the word representations from each layer into the next; in this case we denote the models ATTN-1L, ATTN-2L, etc.

We also analyze multi-head attention models, drawing from (Vaswani et al., 2017). For a model with k heads, we construct one matrix of the form $A \in \mathbb{R}^{d_x/k}$ for each head, and transform each $\mathbf{x}_i$, $\mathbf{x}_i^{k'} = A^{k'}\mathbf{x}_i$ for $k' \in \{1, ..., k\}$. These are passed into the attention equations above, generating context vectors $\mathbf{c}_i^1, ..., \mathbf{c}_i^k \in \mathbb{R}^{d_x/k}$, which are then concatenated to form a vector in $\mathbb{R}^{d_x}$. These context layers are usually sent through a linear transformation to combine features between the heads, the output of which is $\mathbf{c}_i$, but we found that omitting this combination transformation leads to significantly improved results, so we do so in all experiments. We denote these models K-HEAD ATTN.

4.2.1 Relative position representations

We found in early experiments that the absolute position embeddings in self-attention models are insufficient for representing order. Hence, in all attention models except when explicitly noted, we use relative position representations as follows. We follow Shaw et al. (2018), who improved the absolute position representations of the Transformer model (Vaswani et al., 2017) by learning vector representations of relative positions and incorporating them into the self-attention mechanism as follows:

$$\mathbf{c}_i = \sum_{i'=1, j\neq i}^{T} \alpha_j (\mathbf{x}_j + \mathbf{v}_{f(i,j)}) \tag{14}$$

$$\alpha_j = \frac{\exp(\mathbf{x}_i^T W^{(5)} \mathbf{x}_j + b_{f(i,j)})}{\sum_{j'=1}^{T} \exp(\mathbf{x}_i^T W^{(5)} \mathbf{x}_{j'} + b_{f(i,j)})} \tag{15}$$

where $\mathbf{v}_{f(i,j)}$ is a learned vector representing how the relative positions i and j should be incorporated, and $b_{f(i,j)}$ is a learned bias that determines how the relative position should affect the weight given to position j when contextualizing position i. The function f determines which relative positions to group together with a single relative position vector. Given the generally small datasets in IC+SL, we use the following relative position function, which buckets relative positions together in exponentially larger groups as distance increases, following the results of Khandelwal et al. (2018), that LSTMs represent position fuzzily at long relative distances.

$$f(i,j) = \begin{cases} \pm 1\,, |j-i| = 1 \\ \pm 2\,, |j-i| \in \{2,3\} \\ \pm 3\,, |j-i| \in \{4..7\} \\ \ldots \end{cases} \tag{16}$$

This is similar to the preprint of Bilan and Roth (2018), who use linearly increasing bucket sizes; we found exponentially increasing sizes to work well compared to the constant bucket sizes of Shaw et al. (2018).

4.3 Convolutional models

Convolution incorporates local word context into word representations, where kernel width parameter specifies the total size (in words) of local context considered. Each convolutional layer produces a vector representation of each word,

$$\mathbf{h}_{1:T} = \text{relu}(\sqrt{.5} * [\text{CNN}(\mathbf{x}_{1:T}) + \mathbf{x}_{1:T}]) \tag{17}$$

and includes a residual connection, and variance normalization, following (Gehring et al., 2017). To maintain the dimensionality of $\mathbf{h}_i$ as $\mathbb{R}^{d_x}$, we use a filter count of d_x. We vary the number of CNN layers as well as the kernel width, and for all models use a variant known as dilated CNNs. These CNNs incorporate distant context into word representations by skipping an increasing number of nearby words in each subsequent convolutional pass. We use an exponentially increasing dilation size; in the first layer, words of distance 1 are incorporated; at layer two, words of distance 2, then 4, etc. This permits large contexts to be incorporated into word representations while keeping kernel sizes and the number of layers low.

4.4 Recurrent models

We also construct a recurrent word contextualization model, more or less identical to encoders of recent state-of-the-art models. We use a bidirectional LSTM to encode word contexts, $\mathbf{h}_{1:T} =$ BiLSTM$(\mathbf{x}_{1:T})$. As with all other models, we report the performance of this model with feed-forward slot label prediction as well as with label-recurrent slot label prediction. Though similar to earlier work, both models are new; though the latter is recurrent both in word contextualization and slot label prediction, it is distinct from past models in that the two recurrent components are completely decoupled until the prediction step.

5 Datasets

We evaluate our framework and models on the ATIS data set (Hemphill et al., 1990) of spoken airline reservation requests and the Snips NLU Benchmark set (Coucke et al., 2018). The ATIS training set contains 4978 utterances from the ATIS-2 and ATIS-3 corpora; the test set consists of 893 utterances from the ATIS-3 NOV93 and DEC94 data sets. The number of slot labels is 127, and the number of intent classes is 18. Only the words themselves are used as input; no additional tags are used.

The Snips 2017 dataset is a collection of 16K crowdsourced queries, with about 2400 utterances per each of 7 intents. These intents range from 'Play Music' to 'Get Weather'. Training data contains 13784 utterances and the test data consists of 700 utterances. The utterance tokens are mixed case unlike the ATIS dataset, where all the tokens are lowercased. Total number of slot labels are 72. We use IOB tagging, and split 10% of the train set off to form a development set. Utterances in Snips are, on average, short, with 9.15 words per utterance compared to ATIS' 11.2. However, slot label sequences themselves are longer in Snips, averaging 1.8 tokens per span to ATIS' 1.2, making span-level slot labeling more difficult. For our development experiments, we use the casing and tokenization provided by Snips. Co, but to compare to prior work, in one test experiment we use the lowercased, tokenized version of (Goo et al., 2018)[2].

6 Experiments

We evaluate multiple models from each of our model paradigms to help determine what modeling structures are necessary for SLU, and where the best accuracy-speed tradeoffs are. First, we report extensive evaluation across the Snips and ATIS development sets, tracking inference speed and time to convergence along with the usual IC

[2]https://github.com/MiuLab/SlotGated-SLU

Model	label recurrent	IC acc		SL F1		Inference ms/utterance	Epochs to converge	s/epoch	#
		Snips	ATIS	Snips	ATIS				
FEED-FORWARD	No	**98.56**	97.14	53.59	69.68	0.61	48	1.82	17k
FEED-FORWARD	Yes	98.54	**97.46**	**75.35**	**88.72**	1.82	83	2.52	19k
CNN, 5KERNEL, 1L	No	98.56	98.40	85.88	94.11	0.82	23	1.90	42k
CNN, 5KERNEL, 3L	No	99.04	**98.42**	92.21	96.68	1.37	55	2.16	91k
CNN, 3KERNEL, 4L	No	98.81	98.32	91.65	96.75	1.28	57	2.29	76k
CNN, 5KERNEL, 1L	Yes	98.85	98.36	93.12	96.39	2.13	51	2.77	43k
CNN, 5KERNEL, 3L	Yes	**99.10**	98.36	**94.22**	**96.95**	2.68	59	3.34	93k
CNN, 3KERNEL, 4L	Yes	98.96	98.32	93.71	**96.95**	2.60	53	3.43	78k
ATTN, 1HEAD, 1L, NO-POS	No	98.50	97.51	53.61	69.31	1.95	25	1.94	22k
ATTN, 1HEAD, 1L	No	98.53	97.74	75.55	93.22	4.75	117	4.34	23k
ATTN, 1HEAD, 3L	No	**98.74**	98.10	81.51	94.07	7.68	160	4.32	33k
ATTN, 2HEAD, 3L	No	98.31	98.10	83.02	94.61	7.86	79	4.87	47k
ATTN, 1HEAD, 1L, NO POS	Yes	98.63	97.68	74.94	88.60	3.24	60	2.66	24k
ATTN, 1HEAD, 1L	Yes	98.61	98.00	86.72	94.53	6.12	89	5.53	24k
ATTN, 1HEAD, 3L	Yes	98.51	**98.26**	88.04	94.99	9.03	109	6.06	34k
ATTN, 2HEAD, 3L	Yes	98.48	**98.26**	**89.31**	**95.86**	9.17	93	6.54	49k
LSTM, 1L	No	**98.82**	98.34	91.83	97.28	2.65	45	2.91	47k
LSTM, 2L	No	98.77	98.20	93.10	97.36	4.72	58	5.09	77k
LSTM, 1L	Yes	98.68	**98.36**	93.83	**97.37**	3.98	54	4.62	49k
LSTM, 2L	Yes	98.71	98.30	**93.88**	97.28	6.03	69	6.82	79k

Table 1: Development results on the Snips 2017 and ATIS datasets, comparing models from feed-forward, convolutional, self-attention, and recurrent paradigms, as well as comparing non-recurrent, label-recurrent, and fully recurrent architectures, on IC, SL, inference speed, and training time. Inference speed, convergence time, and parameter count are drawn from Snips experiments, but the trends hold on ATIS. The best IC and SL for each dataset is bolded within each model paradigm to help compare between paradigms.

accuracy and SL F1. Second, we pick a small number of our best-performing models to evaluate on ATIS and Snips test sets, to compare against prior work.

For each experiment below, we train until convergence, where convergence is defined by an early stopping criterion with a patience of 30 epochs and an average of development set IC accuracy and token-level SL F1 used as the performance metric.

6.1 Modeling study experiments

In our first category of experiments, we evaluate variants of each word contextualization paradigm introduced.

We evaluate one feed-forward word contextualization module (labeled as FEED-FORWARD) to provide a baseline performance. As with all subsequent models, we evaluate this word contextualization module with and without our proposed label-recurrent decoder. This baseline should help us determine the extent to which each dataset requires the modeling of context.

We evaluate 3 convolutional word contextualization modules. The first has 1 layer with a kernel size of 5, and is intended to provide intuition as to whether a relatively large local context can sufficiently model SL behavior. We label this model CNN, 5KERNEL, 1L, and name all other CNN models similarly. The next model has 3 layers with kernel size 5, and is dilated. This model incorporates long-distance context hierarchically, and is shorter and wider-per-layer than the otherwise-similar 3rd CNN model, with 4 layers and kernel size 3.

We evaluate 4 attention-based word contextualization modules. The first is simple, with 1 attention head and 1 layer. Unlike all others we analyze, it does not use relative position embeddings. Thus, this model is word order-invariant except for a simple absolute position embedding. If it improves over FEEDFORWARD, then, it provides strong evidence that semantic information from the context words, irrespective of order, is useful in making tagging decisions. We label this model with the flag NO-POS. To evaluate the utility of relative position embeddings, we also compare a model with 1 head and one layer, labeled ATTN, 1HEAD, 1L. We then test two increasingly complex models, first with 3 layers and 1 head, the second with 3 layers and 2 heads per layer.

We evaluate 2 LSTM-based word contextualization modules; one uses a single LSTM layer, whereas the other stacks a second on top of the first. As with all other models, we test these two models both with independent slot prediction and label-recurrent slot prediction.

6.2 Comparison to prior work

For our second category of experiments, we take a few high-performing models from our analysis and evaluate them on the Snips and ATIS test sets for comparison to prior work. For these models, we report not only the average IC accuracy and SL F1 across random initializations, but also the standard deviation and best model, as most work has not reported average values. We keep all hyperparameters fixed across all experiments, potentially hindering performance but providing a stronger analysis of robustness.

Note on pre-trained contextual word embedding: Although our framework allows easy integration of contextual pre-trained embeddings like BERT (Devlin et al., 2019) and EMLo (Gardner et al., 2017) by replacing the word contextualization component, we exclude them in our experimentation in order to reduce model obfuscation and have fair comparison against baselines.

7 Results and discussion

In this section, we draw from results reported in Table 1, on the development sets of Snips and ATIS. It is easy to see that very little in the way of modeling is necessary for IC task, so we focus our analysis on SL task. We emphasize that ATIS has shorter spans than Snips, averaging 1.2 and 1.8 tokens, respectively, leading to differing modeling requirements.

7.1 Minimal modeling for SLU

By analyzing three simple models - FEED-FORWARD, ATTN-1HEAD-1L-NO-POS, and CNN-5KERNEL-1L - we conclude that explicitly incorporating local features is a useful inductive bias for high SL accuracy. The purely feed-forward model achieves 53.59 SL F1 on Snips, whereas one layer of convolution improves that number to 85.88. The story is similar for ATIS SL. However, a single layer of attention without position information fails to improve over the feed-forward model whatsoever which we believe is due to the order-invariant nature of self-attention. This also emphasizes the fact that focusing on local context is useful inductive prior for SL task.

For each of these simple models, switching from independent slot label prediction to label-recurrent prediction provides large gains on both datasets. We find an approximate 1.3ms/utterance slowdown from using label recurrence across all models. Thus, in terms of accuracy-for-speed, very simple models can achieve much of the results of more expensive models as long as they are label-recurrent and incorporate local context.

7.2 High-performing convolutional models

The larger convolutional models provide very high accuracy while maintaining fast inference and training speeds. In particular, our best CNN model, CNN-5KERNEL-3L, achieves 94.22 SL F1 on Snips, compared to the two-layer LSTM with label-recurrence, which achieves 93.88. The model achieves this modest improvement with over 2x the inference speed, training in under 1/2 the time, and demonstrating even stronger results on the test sets, discussed below.

On ATIS, where utterances are longer but slot label spans are shorter, LSTMs outperform CNNs on the development sets.

7.3 Issues with self-attention

Our strongest self-attention model underperforms CNNs and LSTMs on both Snips and ATIS, with a maximum SL of 89.31 and 95.86 on the datasets, respectively. Though self-attention models have seen success in complex tasks with lots of training data, we suggest in this study that they lack the inductive biases to perform well on these small datasets.

Relative position embeddings go a long way in improving self-attention models; adding them to a 1-layer attentional encoder improves ATIS and Snips SL by approximately 24 and 22 points, respectively. We find that adding attention heads does not add considerably to the computational complexity of attention models, while increasing accuracy; thus in a speed-accuracy tradeoff, it is likely better to add heads rather than layers as each layer adds $O(n^2 * d_x)$ additional computations.

7.4 Word and label recurrence in LSTMs

Our LSTM word contextualization modules show that with recurrent word context modeling, label-recurrence is less important. For instance, 2-layer LSTM achieves only .78 increase in SL with label recurrence over independent prediction.

		Snips			
		IC Acc		SLR F1	
Model	Recurrence	Mean	Max	Mean	Max
16 LSTM* (Hakkani-Tür et al., 2016)	full	96.9	-	87.3	-
'16 seq2seq+attn* (Liu and Lane, 2016)	full	96.7	-	87.8	-
LSTM+attn+gates (Goo et al., 2018)	full	97.0	-	88.8	-
OUR CNN, 5KERNEL, 3L	none	**97.65±0.28**	97.57	89.57±0.54	90.66
OUR CNN, 5KERNEL, 3L	label	97.57±0.41	**98.29**	**92.30±0.40**	**93.11**
OUR LSTM, 2L	word	97.28±0.36	97.57	90.66±0.55	91.53
OUR LSTM, 2L	full (decoupled)	97.22±0.32	97.14	91.53±0.50	92.62

Table 2: Test set results on the Snips dataset. (*) indicates numbers reported by (Goo et al., 2018)

		ATIS			
		IC Acc		SLR F1	
Model	Recurrence	Mean	Max	Mean	Max
LSTM+attn+gates (Goo et al., 2018)	full	94.10	-	95.20	-
'18 Two LSTMs (Wang et al., 2018)	full	-	**98.99**	-	**96.89**
'18 self-attn+LSTM (Li et al., 2018)	full	-	**98.77**	-	**96.52**
OUR CNN, 5KERNEL, 3L	none	97.04±0.62	97.98	94.84±0.22	94.95
OUR CNN, 5KERNEL, 3L	label	**97.37±0.57**	**98.10**	**95.27±0.19**	**95.54**
OUR LSTM, 2L	word	96.84±0.49	97.65	95.13±0.29	95.41
OUR LSTM, 2L	full (decoupled)	97.00±0.44	97.98	95.15±0.25	95.21

Table 3: Test set results on the ATIS dataset, compared to recent recurrent models.

7.5 Best models compared to prior work

We report test set results on Snips and ATIS in Tables 2 and 3. Our best models from our validation study, CNN-5KERNEL-3L and LSTM-2L, outperform the state-of-the-art on the Snips dataset, with label-recurrence proving crucial, especially for Snips. In particular, CNN-5KERNEL-3L with label recurrence achieves an average SL F1 of 92.30, improving over the previous state-of-the-art of 88.8, by reducing error rate by 30%, and .57-point improvement on IC.

On ATIS, our label-recurrent models outperform slot-gated LSTM model of Goo et al. (2018) on both IC and SL tasks.[3] Wang et al. (2018) attribute their result to using IC and SL-specific LSTMs and use 300-dimensional word embedding and 200-dimensional LSTMs, but with an ATIS vocabulary of 867 words (suggesting a relatively simple sequence space), we are unable to determine the source of the improvement from a modeling standpoint. Similar observation was made for (Li et al., 2018) where 264-dimension embeddings is used.

We hypothesize that our models perform better on Snips because much of Snips slot labeling depends on consistency within long spans, whereas ATIS slot labels have longer-distance dependencies, for example between `to_city` and `from_city` tags.

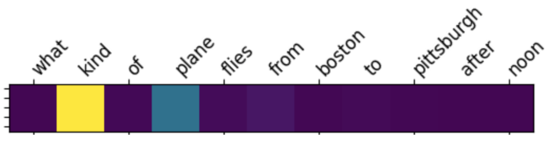

Figure 2: Visualization of the weight given to each token representation by the attention-based pooling for sentence representation. Lighter colors indicate greater attention.

7.6 Attention Visualization

We note that anecdotally, few words in each utterance are useful in indicating the intent. In the example given in Figure 2, presence of possible departure and arrival cities may be distracting, but the attention mechanism correctly learns to focus on words that indicate `atis_aircraft` intent.

8 Conclusion

We presented a general family of joint IC+SL neural architectures that decomposes the task into modules for analysis. Using this framework, we conducted an extensive study of word contextualization methods (including utility of recurrence in the representation and output space) and determined that label-recurrent models, with non-recurrent word representation and a recurrent model of slot label dependencies, are a good fit for

[3]We note that, since this work was performed, considerable efforts have been put into the Snips dataset, including the use of ELMo (Siddhant et al., 2019), BERT (Chen et al., 2019b), and capsule networks (Zhang et al., 2019), among other methods (Chen et al., 2019a).

high performance in both accuracy and speed.

With the results of this study, we proposed a convolution-based joint IC+SL model for SLU that achieves new state-of-the-art results on Snips dataset while maintaining a simple design, shorter training, and faster inference than comparable recurrent methods.

9 Implementation details

All models are implemented in MXNet (Chen et al., 2015). For all models, we randomly initialize word embeddings and use $d_x = 70$. We optimize using Adadelta algorithm (Zeiler, 2012), with initial learning rate, .01. We clip and pad all training and development sentences to length 30, with clipping affecting a small number of utterances. Dropout (Srivastava et al., 2014) probability of .3 is used in all models. We train using a batch size of 128 split across 4 GPUs on a p3.8xlarge EC2 instance, and perform inference using CPUs on same machine.

Acknowledgements

We would like to thank the reviewers for their comments, as well as the Lex team at Amazon AI for helpful conversations and feedback.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *ICLR*.

Ivan Bilan and Benjamin Roth. 2018. Position-aware self-attention with relative positional encodings for slot filling. *arXiv preprint arXiv:1807.03052*.

Mengyang Chen, Jin Zeng, and Jie Lou. 2019a. A self-attention joint model for spoken language understanding in situational dialog applications. *arXiv preprint arXiv:1905.11393*.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019b. BERT for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. In *LearningSys at Neural Information Processing Systems 2015*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A deep semantic natural language processing platform.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. In *ICML*, pages 1243–1252.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 753–757.

Allen L Gorin, Giuseppe Riccardi, and Jeremy H Wright. 1997. How may i help you? *Speech communication*, 23(1-2):113–127.

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

Patrick Haffner, Gokhan Tur, and Jerry H Wright. 2003. Optimizing svms for complex call classification. In *ICASSP*, volume 1, pages I–I. IEEE.

Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. 2016. Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM. In *Interspeech*, pages 715–719.

C.T. Hemphill, J.J. Godfrey, and G.R. Doddington. 1990. The ATIS spoken language systems pilot corpus. In *Proceedings of the DARPA Speech and Natural Language Workshop*, page 96–101.

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

Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context. *Association for Computational Linguistics (ACL)*.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2676–2686. Association for Computational Linguistics.

Changliang Li, Liang Li, and Ji Qi. 2018. A self-attentive model with gate mechanism for spoken language understanding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3824–3833.

B. Liu and I. Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. In *Proceedings of Interspeech*.

Robert E Schapire and Yoram Singer. 2000. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2-3):135–168.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 464–468. Association for Computational Linguistics.

T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang. 2017. DiSAN: Directional self-attention network for RNN/CNN-free language understanding.

Aditya Siddhant, Anuj Goyal, and Angeliki Metallinou. 2019. Unsupervised transfer learning for spoken language understanding in intelligent agents.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

E. Strubell, P. Verga, D. Belanger, and A. McCallum. 2017. Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of EMNLP*.

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.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018. A bi-model based RNN semantic frame parsing model for intent detection and slot filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 309–314.

P. Xu and R. Sarikaya. 2013. Convolutional neural network based triangular CRF for joint intent detection and slot labeling. In *Proceedings of IEEE ASRU Workshop*.

Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. 2014. Spoken language understanding using long short-term memory neural networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 189–194. IEEE.

Fisher Yu and Vladlen Koltun. 2015. Multi-scale context aggregation by dilated convolutions. In *ICLR*.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and Philip S Yu. 2019. Joint slot filling and intent detection via capsule neural networks.

Xiaodong Zhang and Houfeng Wang. 2016. A joint model of intent determination and slot filling for spoken language understanding. In *IJCAI*, pages 2993–2999.