Explainable Slot Type Attentions to Improve Joint Intent Detection and Slot Filling

Kalpa Gunaratna[†], **Vijay Srinivasan**[†], **Akhila Yerukola**[‡*] and **Hongxia Jin**[†]
[†]Samsung Research America, Mountain View CA, USA
{k.gunaratna, v.srinivasan, hongxia.jin}@samsung.com
[‡]Carnegie Mellon University, Pittsburgh PA, USA
ayerukol@andrew.cmu.edu

Abstract

Joint intent detection and slot filling is a key research topic in natural language understanding (NLU). Existing joint intent and slot filling systems analyze and compute features collectively for all slot types, and importantly, have no way to explain the slot filling model decisions. In this work, we propose a novel approach that: (i) learns to generate additional *slot type specific features* in order to improve accuracy and (ii) provides *explanations* for slot filling decisions for the first time in a joint NLU model. We perform an additional constrained supervision using a set of binary classifiers for the slot type specific feature learning, thus ensuring appropriate attention weights are learned in the process to explain slot filling decisions for utterances. Our model is *inherently explainable* and does not need any post-hoc processing. We evaluate our approach on two widely used datasets and show accuracy improvements. Moreover, a detailed analysis is also provided for the exclusive slot explainability.

1 Introduction

Natural language understanding (NLU) is a critical component in building intelligent interactive agents such as Amazon Alexa, Google Assistant, Microsoft's Cortana, and Samsung's Bixby. It helps to understand the intricate details of user utterances, including: (i) understanding the intents of the utterances that help determine the agent's actions and (ii) detecting slots that signify important entities and phrases required to complete the actions. Figure 1 shows an example utterance with an intent and two slots (a slot is a type and value pair). Typically, slot classification (i.e., slot filling) is viewed as a sequence labeling problem which is handled through BIO sequence labeling notation as shown in Figure 1.

State-of-the-art techniques in NLU jointly optimize both the intent detection and slot filling tasks (Chen et al., 2019), including various attention, gating, and transformer-based architectures (Liu and Lane, 2016; Qin et al., 2021). However, existing techniques treat all slot types collectively. For example, they may compute a general attention weights vector to represent all the slot types. Therefore, they do not learn slot type specific patterns, and features based on those patterns that can potentially increase the model performance. In this work, we show that both learning to capture slot type specific patterns through attention weights and computing new features from them 'explicitly' is beneficial in at least two ways: (i) boosting joint NLU model *accuracy*, and (ii) making the model *explain* its slot filling decisions at a granular level. To properly train our slot type specific attention weights learning, we introduce an auxiliary network of binary classifiers to the joint model that has an additional optimization objective. Auxiliary networks and respective optimization objectives in NLP are often used to enforce additional constraints (Joshi et al., 2020; Zhao et al., 2020). We use an auxiliary network to enforce slot type specific attention learning.

On the other hand, explainable AI has gained attention in NLP (Danilevsky et al., 2020) as it provides 'explanations' for 'black box' deep learning model decisions. The terms 'interpretable AI' and 'explainable AI' have been used interchangeably in the literature (Danilevsky et al., 2020). To be consistent and avoid confusion [1], we use the term 'explainable' in this work to refer to a model's abil-

Utterance	Find	a	flight	from	Los	Angeles	to	New	York
BIO labels	O	O	O	O	B-origin	I-origin	O	B-destination	I-destination
Slots	Slot1<type:origin, value:Los Angeles>, Slot2<type:destination, value:New York>								
Intent	FindFlight								

Figure 1: Intent detection and slot filling example.

[*]Work done while at Samsung Research America

[1]Some literature consider *interpretability* to be harder and requires achieving human cognition/performance/agreement.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 3367–3378
December 7-11, 2022

ity to provide explanations to the outside world of its inner working, similar to (Wiegreffe and Pinter, 2019). According to (Jacovi and Goldberg, 2020), explainable NLP models fall into two main categories: (i) models that require external post-hoc processing for explainability, and (ii) models that are inherently explainable due to the capability built into the model. The latter is argued to be preferred in the literature (Rudin, 2019). Explainable models are transparent (Lipton, 2016) and hence, can gain trust. However, existing joint NLU approaches disregard this important explainability aspect. Our proposed joint NLU model in this work is inherently explainable and explains slot filling decisions for each slot type, which is at a granular level. To the best of our knowledge, our approach is the first to inherently support fine-grained explainability for slot filling in a joint NLU model. Our contributions in this work are two-fold:

1. **Accuracy Improvement:** We model per slot type fine-grained attentions and compute new features in addition to intent and slot features to *improve* accuracy of joint intent detection and slot filling task. This is enforced through an auxiliary network of binary classifiers constrained to the number of slot types and supervised using automatically generated data.

2. **Slot Explainability:** We show that per slot type fine-grained attention weights computed through the supervision of the auxiliary network can be used to *explain* the slot filling task in the joint NLU model.

We evaluate our approach using ATIS and SNIPS benchmark datasets and show improvements. We also provide a detailed analysis on the versatility and robustness of slot type specific attention weights in providing insights into slot filling.

2 Related Work

Intent detection and slot filling tasks are two typical sub-tasks of NLU. Slot filling is generally challenging as decisions have to be made for each word/token and can be seen applied in interesting use cases (Gunaratna et al., 2021). The two tasks were performed independently in the past (Haffner et al., 2003; Raymond and Riccardi, 2007). However, jointly optimizing them has shown better accuracy recently (e.g., (Goo et al., 2018; Chen et al., 2019; Qin et al., 2019; Hakkani-Tür et al., 2016; Qin et al., 2021)). Moreover, recent advances in contextual language models have improved the language encoding capabilities that are directly reflected in joint NLU models (Chen et al., 2019; Qin et al., 2021) compared to traditional static word embedding approaches (Hakkani-Tür et al., 2016). Existing joint NLU models leverage RNN (Liu and Lane, 2016; Goo et al., 2018), CNN (Xu and Sarikaya, 2013), intent influenced attention (Dao et al., 2021), gated attention (Goo et al., 2018), self attention (Li et al., 2018; Zhang et al., 2020; He et al., 2020), stack propagation (Qin et al., 2019), hierarchical information flow (Lee et al., 2018; Zhang et al., 2019, 2020), multi task learning (Pentyala et al., 2019), memory networks (Liu et al., 2019), syntactic information integration (Wang et al., 2021), and custom transformer architectures (Qin et al., 2021).

The models mentioned above can learn correlations and hierarchy that exist between intents and slots, but compute features collectively over all slot types. Therefore, these correlations or attention mechanisms are not useful to explain slot filling decisions because (i) they are abstract and do not correspond to each individual slot filling decision, and (ii) may focus on general language semantics or special tokens (Clark et al., 2019). Also, most deep neural models are 'black boxes' and hence, not transparent and explainable (Guidotti et al., 2018), making them hard to earn trust and fix errors. Hence, explainable systems have gained attention for NLP problems such as sentiment classification and question answering (Danilevsky et al., 2020) and outside NLP (Rai, 2020; Zhang and Chen, 2020). Importantly, Rudin argues that developing inherently explainable models is important because use of post-hoc techniques to explain existing systems leads to many problems (Rudin, 2019). Ours is the first inherently explainable method for slot filling in a joint NLU model.

In our proposed model, we compute slot type specific attention weights to explain slot filling decisions. There has been a debate on whether attentions are explainable (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019). However, it has been shown that attentions can be used for explanations, if they contribute towards the model outcomes/behaviors (Wiegreffe and Pinter, 2019). We show that this is true in our model and provide evidence for a strong correlation between the slot type attentions and slot predictions (Section 4.2).

Orthogonal to ours, having multiple final slot classifiers is possible to model separate features but it introduces the big challenge of resolving overlapping spans (i.e., nested NER) (Wang et al., 2020).

3 Approach

Our approach consists of three network components: (i) intent classifier, (ii) slot classifier, and (iii) slot type weight and feature generator. Existing joint NLU models consist of the first two network components arranged in various configurations whereas ours is distinguishably different from them because of the third auxiliary network component. This third component learns to compute: (i) separate per slot type features (i.e., logits) and (ii) fine-grained per slot type attention weights for each utterance token. Further, it is supervised through automatically generated training data (see Section 3.1.2) and constrained to exactly the number of slot types. Our per slot type attention weights can provide insights on the focus points of the input utterance that are relevant to labeling a particular slot type, which is absent in other approaches. Our approach is illustrated in detail in Figure 2.

3.1 Network

Let l be the number of tokens in the input utterance u=[t_1, t_2, t_3, ..., t_l] where t_i represents the ith token in u. Let T be the original slot type set in the ground truth (e.g., location, time, etc.), S be the set of sequence labeling format (BIO) labels (e.g., O, B-location, I-location, etc.), and I be the set of intents in the ground truth (e.g., FindFlight, Search, etc.). It is important to model dynamics of O (in BIO) in our model and hence, O $\in T$.

Given an utterance u, we first tokenize and encode it using a contextual language model like BERT. Then we get the encoded utterance u^e=[$\boldsymbol{t_1}, \boldsymbol{t_2}, \boldsymbol{t_3}, ..., \boldsymbol{t_l}$] ($\in \mathbb{R}^{l \times d}$) where token $\boldsymbol{t_i}$ represents the embedding vector for the ith token, $\boldsymbol{t_i} \in \mathbb{R}^d$ and d is the embedding dimension.

3.1.1 Intent Classifier

Intent classifier is a single layer multi-class classifier that uses BERT as the language encoder. For a given utterance u, the embedding vector for [CLS] token is the context embedding $u^c \in \mathbb{R}^d$. Intent classifier outputs intent logits g^{intent} as follows.

$$g^{intent} = u^c W^{intent} + b^{intent} \quad (1)$$

where, $W^{intent} \in \mathbb{R}^{d \times |I|}$ is the learnable weight matrix and b^{intent} is the bias. Y^{intent} is the one hot encoded intent vector and $p^{intent} = softmax(g^{intent})$. Intent loss $\mathcal{L}^{intent}$ for utterance u is computed using cross entropy loss as follows.

$$\mathcal{L}^{intent} = -\sum_{x}^{|I|} Y_x^{intent} log(p_x^{intent}) \quad (2)$$

3.1.2 Slot Type Weight and Feature Generator

We discuss details of our auxiliary network that helps the model learn explainable weights and generate general yet slot type specific features. First, after applying dropout to the encoded utterance u^e, we concatenate intent logits g^{intent} with u^e. Note that $g^{intent} \in \mathbb{R}^{|I|}$, hence we expand the logits vector by one dimension, copy the values along the expanded dimension l times (utterance length) to get the expanded intent logits tensor g^{int_e} ($\in \mathbb{R}^{l \times |I|}$). Concatenation of intent logits with token embeddings and then applying self attention provides direct fusion of intent features with the slot classification network to learn any correlations between intents and slots. We compute the new feature u^{sa} ($\in \mathbb{R}^{l \times d}$) as follows. LL, LN, and SA represent linear layer, layer normalization, and self attention functions, respectively. $\oplus$ is the concatenation.

$$u^{sa} = SA(LL(LN(u^e \oplus g^{int\_e}; \theta^{LN}); \theta^{LL}); \theta^{SA}) \quad (3)$$

θ^{LN}, θ^{LL}, and θ^{SA} are the sets of model parameters. Self attention function $SA(.)$ is as follows.

$$SA(x; \theta^{SA}) = softmax\left(\frac{Q_x K_x^T}{\sqrt{d_x}}\right) V_x \quad (4)$$

Query Q_x, key K_x, and value V_x are calculated from input x using three different linear projections, $LL_q(x)$, $LL_k(x)$, and $LL_v(x)$, respectively. d_x is the dimension of the input x (also K_x). In Equation 3, LL projects back to dimension d and $d_x = d$ in SA. Then we add the original utterance embedding u^e to u^{sa} and perform a layer normalization to get $\hat{u^e}$ before the auxiliary network.

$$\hat{u^e} = LN(u^e + u^{sa}; \theta) \quad (5)$$

Slot type constrained self attention weights

We project the utterance embedding $\hat{u^e}$ into multiple representations; exactly $|T|$ times. We want our model to explain slots at a fine-grained level

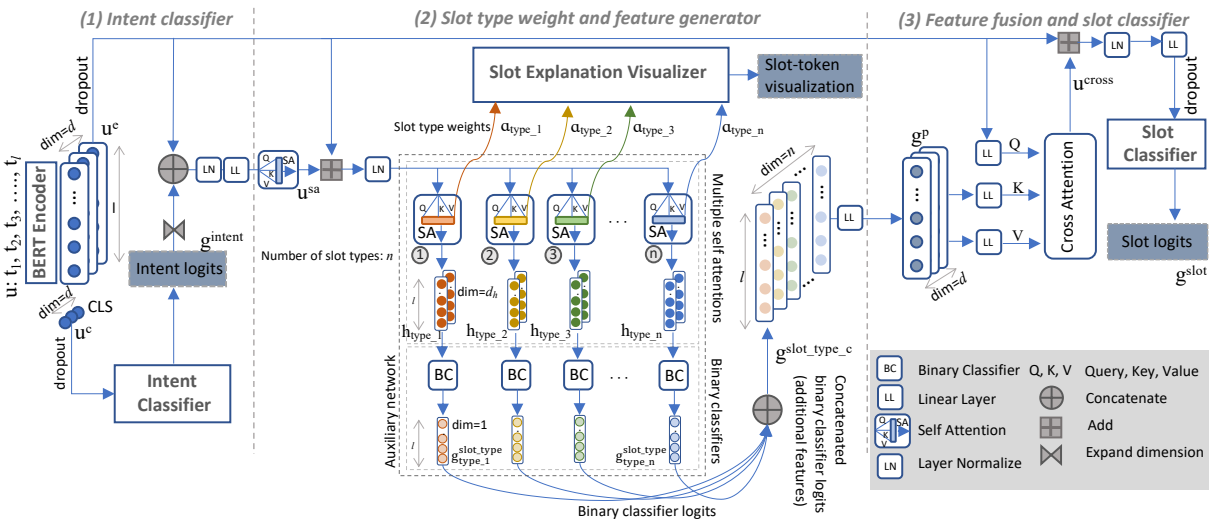

Figure 2: Overview of our explainable joint NLU model. n is the total number of slot types, l is the utterance length, d_h is the slot type converted utterance embedding dimension, and d is the original utterance embedding dimension. In addition to intent and slot predictions, our model visualizes per slot type attentions as explanations. We model per slot type features and learn appropriate weights - different colors show each slot type modeling.

and hence, we need weight distributions over the entire input for each token, for each slot type. We compute multiple self attentions, one per slot type, that provides slot type specific attention weights per token with respect to the utterance.

However, without proper supervision, these attention weights are not meaningful. Therefore, we make these multiple projections correspond to $|T|$ slot type binary classifiers. We apply the self attention defined in Equation 4, $|T|$ number of times, one for each slot type by projecting different query Q_{type_i}, key K_{type_i}, and value V_{type_i} tensors ($\in \mathbb{R}^{l \times d_h}$) from $\hat{u^e}$, each using three different linear projections. Self attended output h_{type_i} ($\in \mathbb{R}^{l \times d_h}$) and attention weights α_{type_i} ($\in \mathbb{R}^{l \times l}$) for slot type $type\_i$ ($\in T$) are as follows. d_h is the embedding dimension.

$$h_{type\_i} = softmax\left(\frac{Q_{type\_i}K^T_{type\_i}}{\sqrt{d_h}}\right)V_{type\_i}$$

$$\alpha_{type\_i} = softmax\left(\frac{Q_{type\_i}K^T_{type\_i}}{\sqrt{d_h}}\right) \quad (6)$$

Slot type feature generation and supervision

Slot type projections require additional optimization objectives so that the self attention weights are truly meaningful and not random noise. We train each slot type projection to predict binary output that states whether an input token in the utterance

Utterance : "Find a flight from Los Angeles to New York"
Slot type set (T) : {O, origin, destination, time}
Seq. label set (S) : {O, B-origin, I-origin, B-destination, I-destination, B-time, I-time}

Utterance		Find	a	flight	from	Los	Angeles	to	New	York
Sequence labeling		O	O	O	O	B-origin	I-origin	O	B-destination	I-destination
Binary format	origin	0	0	0	0	1	1	0	0	0
	destination	0	0	0	0	0	0	0	1	1
	time	0	0	0	0	0	0	0	0	0
	O	1	1	1	1	0	0	1	0	0

Figure 3: Data generation for the auxiliary network.

is true (1) or false (0) for that slot type. This training data is automatically generated from sequence labeling BIO ground truth as shown in the example in Figure 3. Binary format for a slot type except for O is generated by having 1s to slot tokens (non-O positions) and 0s to otherwise. For O, all non-slot tokens are marked as 1s and rest are 0s.

With the additional optimization objectives for each slot type, self attention weights can be used to visualize attention points of the input for each token with respect to each slot type as explanations. This is because: (i) per slot type, the embedding for the token being classified is computed attending to all the input tokens including itself and (ii) the same token embedding is used for the particular slot type classification (controlled supervision), and further, (iii) type classification output logits are fed to the final slot classification as additional features (explicit connection to final classification). Our binary classification output logits enable the model to learn general patterns/features specific to each slot type that also boost model performance. Binary classification for ith slot type $type\_i$ ($\in T$) is

performed as follows. We initialize a set of weight tensors W^H and bias b^H where $|W^H| = |b^H| = |T|$.

$$g_{type\_i}^{slot\_type} = h_{type\_i} W_{type\_i}^H + b_{type\_i}^H \quad (7)$$

where $h_{type\_i} \in \mathbb{R}^{l \times d_h}$, $W_{type\_i}^H \in \mathbb{R}^{d_h \times 1}$, $b_{type\_i}^H \in \mathbb{R}$, and slot type logits $g_{type\_i}^{slot\_type} \in \mathbb{R}^{l \times 1}$. Binary cross entropy loss $\mathcal{L}^{type}$ for type classification for utterance u is as follows.

$$\mathcal{L}^{type} = -\frac{1}{N}\left(\sum_{j=1}^{N} Y_j^t log(p_j^t) + (1 - Y_j^t) log(1 - p_j^t)\right) \quad (8)$$

where, Y^t is the one-hot encoded ground truth vector and p^t is the predicted probabilities vector, and N is the total number of data points. Note that we measure binary cross entropy loss collective for all the slot type classifiers. Therefore, if $|u| = l$, then $N = l \times |T|$.

Slot explanation visualizer

We take the attention weights matrix α_{type_i} for a slot type $type\_i$ and visualize attentions on the utterance for each token. See Figure 4 for examples.

3.1.3 Feature Fusion and Slot Classifier

Slot type logits generated from the previous step are new features that go into our slot classifier to improve accuracy. They are slot type specific and hence, can capture fine-grained slot patterns. We concatenate all the slot type logits per each input utterance token and then apply cross attention on it by having query Q_{u^e} projected from the utterance u^e and key K_{g^p} and value V_{g^p} projected from the concatenated and projected slot type logits g^p. The logits concatenation is performed at the token level where all the slot type logits for a token are concatenated to produce the tensor $g^{slot\_type\_c} (\in \mathbb{R}^{l \times |T|})$. Then the concatenated logits tensor is projected to get g^p ($\in \mathbb{R}^{l \times d}$). We then apply cross attention between utterance embedding u^e and g^p to get slot type aware utterance representation u^{cross} ($\in \mathbb{R}^{l \times d}$). We compute query, key, and value tensor projections ($\in \mathbb{R}^{l \times d}$) as follows. θ^1, θ^2, and θ^3 are layer parameters for the three different linear projection layers.

$Q_{u^e} = LL(u^e; \theta^1), K_{g^p} = LL(g^p; \theta^2), V_{g^p} = LL(g^p; \theta^3)$

$$u^{cross} = softmax\left(\frac{Q_{u^e} K_{g^p}^T}{\sqrt{d}}\right) V_{g^p} \quad (9)$$

Cross attention between the utterance embeddings and slot type logits highlights slot type specific features from the utterance embeddings. This is added to utterance as follows to make the slot classifier input u^{slot} ($\in \mathbb{R}^{l \times d}$).

$$u^{slot} = LL(LN(u^e + u^{cross}; \theta^{s_{LN}}); \theta^{s_{LL}}) \quad (10)$$

Finally, the slot logits tensor g^{slot} ($\in \mathbb{R}^{l \times |S|}$) is computed where $W^{slot} \in \mathbb{R}^{d \times |S|}$.

$$g^{slot} = u^{slot} W^{slot} + b^{slot} \quad (11)$$

The slot loss $\mathcal{L}^{slot}$ for utterance u is computed using the cross entropy loss. $Y_{i,s}^{slot}$ is the one hot encoded ground truth slot label and $p_{i,s}^{slot}$ is the softmax probability of slot BIO label s for i^{th} token.

$$\mathcal{L}^{slot} = -\sum_{i}^{l} \sum_{s}^{|S|} Y_{i,s}^{slot} log(p_{i,s}^{slot}) \quad (12)$$

Network optimization

We optimize our network by minimizing the joint loss of the three sub-networks. Overall loss $\mathcal{L}$ for utterance u is defined as in Equation 13. α, β, and γ are hyperparameters that represent loss weights.

$$\mathcal{L} = \alpha \mathcal{L}^{intent} + \beta \mathcal{L}^{type} + \gamma \mathcal{L}^{slot} \quad (13)$$

4 Evaluation and Discussion

We use two most widely used datasets for joint NLU: ATIS (Hemphill et al., 1990) and SNIPS (Coucke et al., 2018). ATIS has 4478, 500, and 893 utterances for train, dev, and test splits, respectively. SNIPS has 13084, 700, and 700 utterances for train, dev, and test splits, respectively. We measure slot accuracy on the exact span match and maximum utterance length is 50.

Hyperparameters We experimented with batch size, and number of epochs and set learning rate and dropout as in JointBert (Chen et al., 2019)(appendix A.3). We experimented with 20, 30, and 40 epochs, batch sizes of 32, 64, and 128, and trained the model for the entirety of the epochs[2]. Using 40 and 20 epochs with batch size 32 gave the best performance on ATIS and SNIPS datasets, respectively. For all the experiments, we used

[2]We saved the model at the end of all training epochs. Training used only the training dataset split; dev data split was used for hyperparameter selection.

Model	SNIPS		ATIS	
	Intent	Slot	Intent	Slot
RNN-LSTM (Hakkani-Tür et al., 2016)	96.9	87.3	92.6	94.3
Attention-BiRNN (Liu and Lane, 2016)	96.7	87.8	91.1	94.2
Slot-Gated (Goo et al., 2018)	97.0	88.8	94.1	95.2
Joint BERT (Chen et al., 2019)	98.6	97.0	97.5	96.1
Joint BERT + CRF (Chen et al., 2019)	98.4	96.7	97.9	96.0
Co-Interactive Transformer (Qin et al., 2021)	98.8	95.9	97.7	95.9
Co-Interactive Transformer + BERT (Qin et al., 2021)	98.8	97.1	98.0	96.1
Ours (BERT as the language encoder)	**98.99**	**97.24**	**99.10**	**96.20**

Table 1: Joint intent detection and slot filling results on the two benchmark datasets. Our novel slot type specific attention model for joint NLU task improves the state-of-the-art model performance.

Model	SNIPS		ATIS	
	Intent	Slot	Intent	Slot
Our Full Model	**98.99**	**97.24**	**99.10**	**96.20**
(-) slot type feature and weight generator	98.85	96.77	98.54	96.00
(-) cross attention	98.71	96.57	99.10	95.76
(-) intent logit concatenation with utterance	98.99	96.44	99.10	95.87

Table 2: Ablation experiments of our approach. (-) refers to removing that part from the network in Figure 2.

learning rate of 5e-5, dropout of 0.1, and Adam optimizer. In our experiments, we set α=β=γ=1 and the slot type network attention (projection) dimension d_h=32. We implemented our approach in PyTorch and used BERT-Base (any encoder works). We use BertViz (Vig, 2019) for visualizations.

4.1 Joint NLU Model Accuracy

We consider several comparable baselines to evaluate against our approach. RNN-LSTM (Hakkani-Tür et al., 2016) uses LSTMs to predict the intent and slot labels whereas Attention-BiRNN (Liu and Lane, 2016) is an extension which uses attention mechanisms. Slot-Gated (Goo et al., 2018) introduced gating mechanism on LSTMs to improve joint intent detection and slot filling. These approaches represent the pre-contextual language model era. The advent of contextual language models such as BERT have enabled significant improvements in NLU models. Joint BERT (Chen et al., 2019) uses BERT as the language encoder, using the [CLS] token for intent classification and token embeddings for slot filling. Co-Interactive Transformer (Qin et al., 2021) uses modified transformer layers after the initial BERT encoding. We also use BERT as the language encoder.

As shown in Table 1, our joint NLU approach with the addition of slot type attentions and auxiliary network yields the best performance. The slot type specific modeling not only improves the slot filling performance but also the intent detection accuracy. We also see in Table 1 that BERT and transformer-based NLU models (Chen et al., 2019; Qin et al., 2021) outperform sequence labeling models based on RNN or LSTM. Our model shows accuracy improvements that go even further, due to its novel slot type specific feature computation and fusion.

We also present an ablation study by removing 3 components from Figure 2 as shown in Table 2. We see that there are significant drops in both the intent and slot task performance when individual components are removed. Additionally, it is interesting to see that dropping the cross attention with the slot type logit features and utterance embedding affects the performance more than dropping the entire slot type auxiliary network. This may be because computing features is not sufficient; appropriately fusing them is necessary and important.

4.2 Slot Explainability

A novel feature of our joint NLU model is slot explainability. Using attentions as explanations has caused some debate in the NLP community (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019). However, (Wiegreffe and Pinter, 2019) [3] argued

[3]They further stated "Attention mechanisms do provide a look into the inner workings of a model, as they produce an

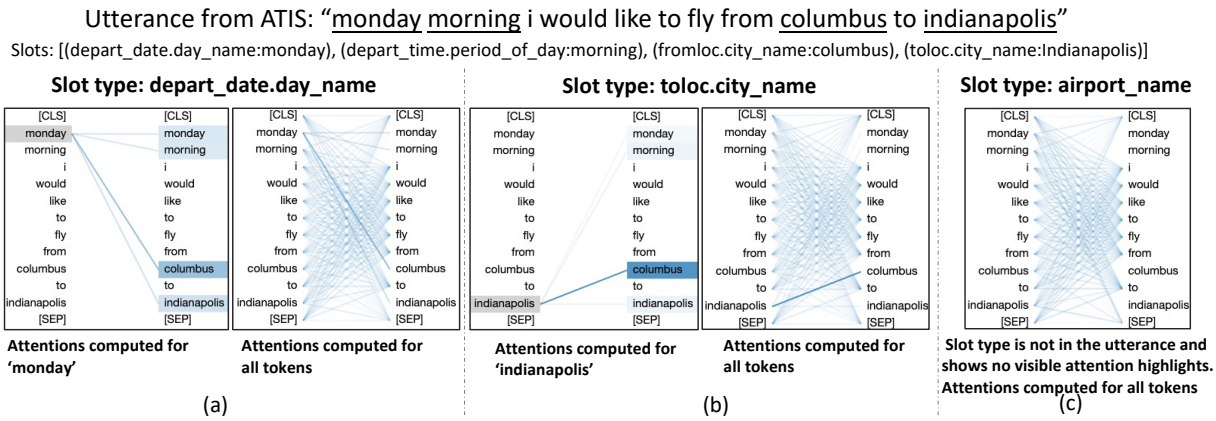

Figure 4: Slot type specific attentions and explainable visualizations on an example utterance from ATIS. (a) and (b) show plots for two 'positive' slot types (that appear in the utterance) whereas (c) shows a plot for a 'negative' slot type. Blue rectangles in the left plots in (a) and (b) show the attention points of the model for the tokens in the gray rectangle and the target slot type.

SNIPS		ATIS	
Intent	Slot	Intent	Slot
98.99	96.88	98.99	95.78

Table 3: Accuracy while freezing slot type attentions (from a uniform distribution).

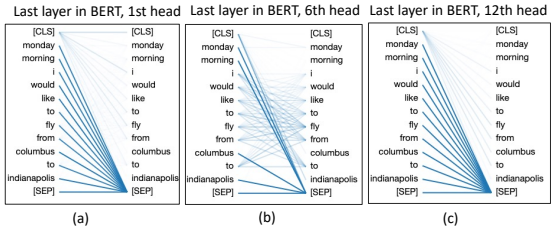

Figure 5: Attentions in the last BERT layer. (a), (b), and (c) shows first, sixth, and twelfth attention heads.

that one can use attentions to provide explanations if they are 'necessary for good performance', and suggested several experiments for this purpose. We present our findings along this line in Table 3 where model training and predictions were performed while freezing slot type attention weights to be a uniform distribution. We can see that then the model performance drops, which implies that these attention weights represent important and meaningful features.

Slot type specific attention weights reflect where the model focuses on the utterances. Figure 4 shows computed attention weights belonging to some slot types when visualized against an example utterance taken from ATIS. Slot types *depart_date.day_name* and *toloc.city_name* are two positive slot types (out of 4) for the utterance. In Figure 4 (a) and (b), we can see where the model is concentrating for the particular slot type. This is only possible because we model per slot type feature computation through self attention (note that simple query-based attentions will not work for explainability), supervision (via auxiliary network), and also fusion in the main network. For example, consider attention weights for *toloc.city_name* slot type, to classify 'indianapolis' as *toloc.city_name*, model pays attention to token 'columbus' which belongs to *fromloc.city_name*. Similarly for the token 'monday' that belongs to type *depart_date.day_name*, the model pays attention to *fromloc.city_name* and *toloc.city_name* positions. In Figure 4(c), we show attention weights computed for the slot type *airport_name* that is not present in the utterance (i.e., negative slot type) and all the attentions computed for it are equally low, with no significant attention points. It is evident from this example that using slot type attentions, we are able to see the general patterns that the model looks at for each slot type.

In fact, explicit modeling and network design is required to provide explanations like we have shown. For example, consider multi-head attention layers in BERT (multiple heads in the 12 layers in BERT base). Despite multiple heads being deployed to learn and compute multiple feature vectors, they do not correlate with the slot types as there is no 1-1 mapping of slot types to attention heads through any supervision; hence, these attention heads cannot be used for slot explanations. Clark et al. (Clark et al., 2019) analyzed BERT layer attentions and they observed that some atten-

easily-understandable weighting of hidden states."

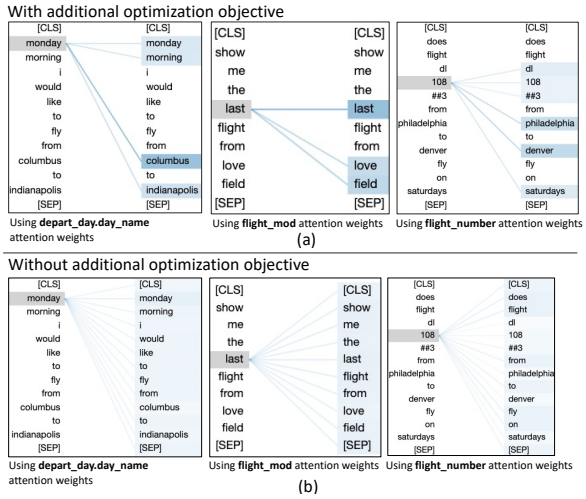

Figure 6: Slot type specific attention weights have to be properly supervised. In these visualizations, (a) shows that our auxiliary network design enables proper attention computations whereas, (b) shows that simply performing multiple attention computations and fusing does not yield proper attentions.

tions mainly focus on general language semantics and special tokens (e.g., [SEP]). In Figure 5, we show several attention heads in the last BERT layer, and we see that they cannot be used to explain slot labeling decisions for each slot type, this is also true for any attention attribution techniques (e.g., (Hao et al., 2021)), self attention or traditional attention methods applied on the utterance since a single attention weight vector cannot be broken down to multiple slot types.

To further illustrate this effect, we removed the auxiliary binary classification (i.e., slot type output optimization) that uses the automatically generated data in the training phase and used the resulting model to visualize attentions for the slot types. See Figure 6(a), where it shows that the network learns proper attention weights for each slot type with our auxiliary network compared to Figure 6(b), where it shows when the additional optimization is removed, network is not guaranteed to learn any meaningful attention weights vector.

4.2.1 Quantitative Analysis

It is important to verify that the observations shown in Figure 4 are consistent. Our hypothesis is that positive slot type attention weights (Figure 4 (a) and (b)) have clear spikes or specific focus areas in attention compared to the negative slot type attention weights (Figure 4 (c)). We use entropy[4] to capture this difference. We expect positive slot type attention weights to have lower entropy compared to the negative slot type attention weights, which tend to have a more uniform distribution, hence, higher entropy. We compute entropy for a list of attention weights $[x_1, x_2, .., x_n]$ using the equation below, where, $Px_i = x_i / \sum x_i$.

Top $k\%$ attn.	SNIPS			ATIS		
	Pos.	Neg.	Diff	Pos.	Neg.	Diff
100%	**6.1161**	6.1933	0.0772	**6.3135**	6.4009	0.0874
10%	**3.8986**	4.0075	0.1089	**3.5281**	3.7878	0.2597
5%	**2.8803**	2.9955	0.1152	**2.4589**	2.7628	0.3039

Table 4: Average entropy for top $k\%$ attention weights for test data. Results confirm attention spikes in positive slot types compared to negative ones. Lower entropy means non-uniform values. Positive slot types (Pos.) are the slot types that appear in an utterance and negative slot types (Neg.) are the ones do not appear.

$$entropy = -\sum Px_i \times log_2(Px_i) \quad (14)$$

We compute the average entropy of the positive slot types and the negative slot types separately for each utterance and then average the entropy values. As shown in Table 4, the positive slot types achieve lower average entropy values compared to the negative slot types for both the datasets. The table also shows a breakdown of top $k\%$ attention weights (sorted in descending order and take the first $k\%$ items). We can see that when k is small, like 5% or 10%, the entropy difference between positive and negative slot types is higher. The attention vectors are long, and hence several attention spikes for positive slot types may not seem prominent in the entropy computation when the entirety of the attention weights is considered. These results suggest that there are more attention spikes in positive slot types for utterances compared to negative slot types. Thus, we can use the distinguishable attention spikes to gain insights into slot filling decisions.

4.2.2 Qualitative Analysis

Our goal is to inspect whether the model attentions are consistent, trustworthy, and not arbitrary. We performed a small-scale human evaluation to check whether slot type attentions remain similar if the

[4] https://en.wikipedia.org/wiki/Entropy_(information_theory)

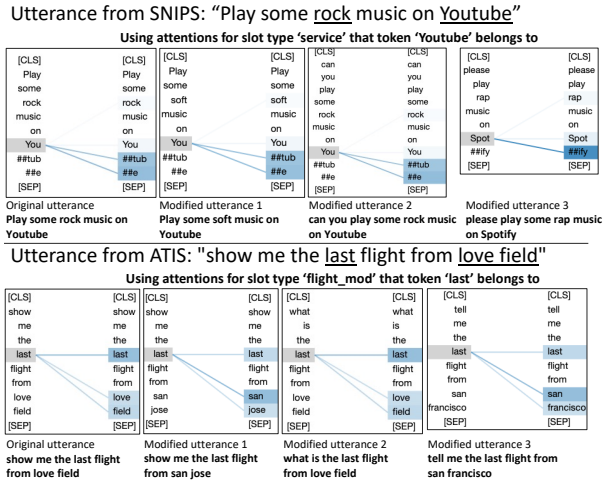

Figure 7: Attention visualizations for modified utterance examples. We can see that the attentions are consistent across modifications, hence, trustworthy.

utterances are slightly modified. We randomly selected 20 utterances each, from ATIS and SNIPS. Three types of modifications were applied to each of these utterances: (i) modifying slot values only, (ii) utterance text only, and (iii) both utterance text and slot values. We obtained a total of 6 modifications for each utterance (2 per modification type). We make sure that utterance semantics (original slots types) and the meaning of the utterances do not change due to these modification. Figure 7 shows few examples of such modifications. We collected 120 modifications (20 x 6) for each dataset and 240 (120 x 2) in total for both datasets.

For each original utterance, a positive slot type was selected at random and the corresponding computed attention patterns were compared with the original utterance for each of the modifications. We used a pool of two judges and they were asked to rate the similarity of the attention patterns on a 3-point Likert scale (3 for exactly same, 2 for similarity with some overlap, 1 for no overlap). We obtained 344/360 for ATIS and 354/360 for SNIPS (the total possible score per dataset is 360=120x3). These scores indicate that the attentions for slot types were consistent across the modifications and they are not arbitrary or opaque. This shows that our slot type specific attention mechanism is able to successfully learn patterns that are specific to slot types 'explicitly'. These patterns are also consistent, and hence, these slot type specific attention weights can provide a means for trustworthy explanations. See Appendix A for few more examples.

5 Conclusion

We presented a novel joint NLU approach where we jointly model intent detection and slot filling with per slot type attention learning. Slot type specific attention weights enabled the model to provide fine-grained slot explanations and additional slot type features computed through the auxiliary network (i.e., logits) improved model accuracy. To the best of our knowledge, our proposed approach is the first to inherently provide explanations for slot filling while jointly detecting both intents and slots in utterances, without any post-hoc processing. The added transparency from our explainable model is beneficial for further debugging and improvement of models, and for increasing end user trust in personal voice assistants and task oriented dialog systems. In future, we plan to investigate the use of this type of inherently explainable model through auxiliary network constraint enforcement for problems such as text classification and NER.

Limitations

Currently, our proposed model provides explanations for the slot filling task, which is challenging because we need to provide explanations for each word in an utterance. It is possible to extend our model to explain intent detection as well, where explanations are given for the classification decision over the entire utterance. We have not yet explored this aspect of explainable intent detection in the joint model. Our approach consists of additional auxiliary network and it needs additional parameters on top of BERT-Base model to properly learn new features and attention weights. For example, for SNIPS dataset, our model uses 120 million trainable parameters (model size is 461MB) compared to 110 million parameters in BERT-Base model (model size is 438MB). Hence, our model is larger than BERT base and the increase in model size will depend on the number of slot types in the dataset. However, in real life, for joint NLU models, the number of slot types is in the order of 10s (less than 100 in most cases) and since we are not considering this type of a model for on-device deployment, we believe this is manageable. Note that number of slot types is around half the size of BIO slot labels.

References

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

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does bert look at? an analysis of bert's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

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*.

Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. 2020. A survey of the state of explainable ai for natural language processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 447–459.

Mai Hoang Dao, Thinh Hung Truong, and Dat Quoc Nguyen. 2021. Intent detection and slot filling for vietnamese. *arXiv preprint arXiv:2104.02021*.

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)*, pages 753–757.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. 2018. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42.

Kalpa Gunaratna, Vijay Srinivasan, Sandeep Nama, and Hongxia Jin. 2021. Using neighborhood context to improve information extraction from visual documents captured on mobile phones. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 3038–3042.

Patrick Haffner, Gokhan Tur, and Jerry H Wright. 2003. Optimizing svms for complex call classification. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, 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.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12963–12971.

Keqing He, Shuyu Lei, Yushu Yang, Huixing Jiang, and Zhongyuan Wang. 2020. Syntactic graph convolutional network for spoken language understanding. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2728–2738.

Charles T Hemphill, John J Godfrey, and George R Doddington. 1990. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205.

Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3543–3556.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Jihwan Lee, Dongchan Kim, Ruhi Sarikaya, and Young-Bum Kim. 2018. Coupled representation learning for domains, intents and slots in spoken language understanding. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 714–719. IEEE.

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.

Zachary C Lipton. 2016. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*.

Bing Liu and Ian Lane. 2016. Attention-based recurrent neural network models for joint intent detection and slot filling. *Interspeech 2016*, pages 685–689.

Yijin Liu, Fandong Meng, Jinchao Zhang, Jie Zhou, Yufeng Chen, and Jinan Xu. 2019. Cm-net: A novel collaborative memory network for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1051–1060.

Shiva Pentyala, Mengwen Liu, and Markus Dreyer. 2019. Multi-task networks with universe, group, and task feature learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 820–830.

Libo Qin, Wanxiang Che, Yangming Li, Haoyang Wen, and Ting Liu. 2019. A stack-propagation framework with token-level intent detection for spoken language understanding. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2078–2087.

Libo Qin, Tailu Liu, Wanxiang Che, Bingbing Kang, Sendong Zhao, and Ting Liu. 2021. A co-interactive transformer for joint slot filling and intent detection. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8193–8197. IEEE.

Arun Rai. 2020. Explainable ai: From black box to glass box. *Journal of the Academy of Marketing Science*, 48(1):137–141.

Christian Raymond and Giuseppe Riccardi. 2007. Generative and discriminative algorithms for spoken language understanding. In *Interspeech 2007-8th Annual Conference of the International Speech Communication Association*.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42.

Jixuan Wang, Kai Wei, Martin Radfar, Weiwei Zhang, and Clement Chung. 2021. Encoding syntactic knowledge in transformer encoder for intent detection and slot filling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13943–13951.

Jue Wang, Lidan Shou, Ke Chen, and Gang Chen. 2020. Pyramid: A layered model for nested named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5918–5928.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20.

Puyang Xu and Ruhi Sarikaya. 2013. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 ieee workshop on automatic speech recognition and understanding*, pages 78–83. IEEE.

Chenwei Zhang, Yaliang Li, Nan Du, Wei Fan, and S Yu Philip. 2019. Joint slot filling and intent detection via capsule neural networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5259–5267.

Linhao Zhang, Dehong Ma, Xiaodong Zhang, Xiaohui Yan, and Houfeng Wang. 2020. Graph lstm with context-gated mechanism for spoken language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9539–9546.

Yongfeng Zhang and Xu Chen. 2020. Explainable recommendation:: A survey and new perspectives. *Foundations and Trends® in Information Retrieval*, 14(1):1–101.

Yufan Zhao, Can Xu, and Wei Wu. 2020. Learning a simple and effective model for multi-turn response generation with auxiliary tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3472–3483.

A Appendix

A.1 Slot type specific attentions visualizations

Here we show two more utterances taken from ATIS and SNIPS and show visualizations for modifications of the original utterances in Figure 8.

A.2 Example utterances in qualitative analysis

Here we show six modifications for the above two randomly selected utterances. The modifications are shown in Table 5.

A.3 Experiment details

We used Tesla V100-SXM 32GB gpu configured in a gpu cluster and used CUDA 10.1. Our model for ATIS has 130 million parameters and for SNIPS, it has 120 million parameters. ATIS model has little more parameters because it has more slot types. Saved model size for ATIS dataset is 500MB whereas for SNIPS, it is 461MB.

Best performing development set accuracy values are as follows. For ATIS, intent accuracy and slot F1 values are 98.60 and 98.33, respectively. For SNIPS, they are 99.14 and 97.60, respectively. On average, training takes around 20 - 25 minutes on the above GPU.

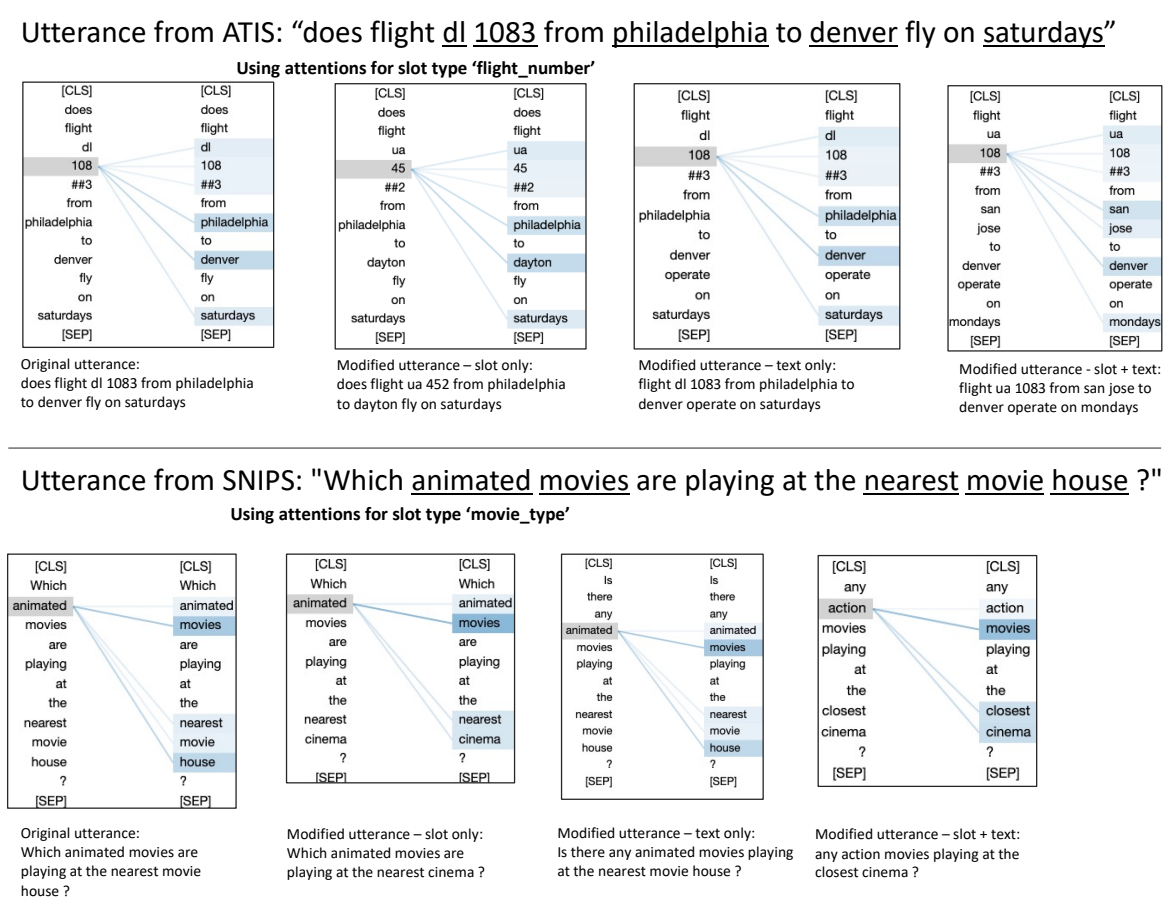

Figure 8: Analyzing attentions for slightly modified utterances where semantics of the modified are the same as originals. Examples show that attentions computed for specific slot types remain very similar.

Dataset	Original Utterance	Modification Type	Modified Utterance
ATIS	does flight dl 1083 from philadelphia to denver fly on saturdays	Slot only	(1) does flight ua 1083 from philadelphia to denver fly on saturdays (2) does flight ua 452 from philadelphia to dayton fly on saturdays
		Text only	(3) flight dl 1083 from philadelphia to denver fly on saturdays (4) flight dl 1083 from philadelphia to denver operate on saturdays
		Slot + Text	(5) flight ua 1083 from philadelphia to denver operate on saturdays (6) flight ua 1083 from san jose to denver operate on mondays
SNIPS	Which animated movies are playing at the nearest movie house ?	Slot only	(1) Which animated movies are playing at the nearest cinema ? (2) Which action movies are playing at the farthest cinema ?
		Text only	(3) What animated movies are playing at the nearest movie house ? (4) Is there any animated movies playing at the nearest movie house ?
		Slot + Text	(5) Is there any action movies playing at the nearest cinema ? (6) any action movies playing at the closest cinema ?

Table 5: Showing two original utterances from ATIS and SNIPS and how each utterance gets modified into 6 unique utterances to test slot type attention weights for consistency.