# INTERPRETABLE SINGLE/MULTI-LABEL TEXT CLASSIFICATION WITH UNSUPERVISED CONSTITUENT-LABEL ALIGNMENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Deep neural networks based on layer-stacking architectures have historically suffered from poor inherent interpretability. Meanwhile, symbolic probabilistic models function with clear interpretability, but how to combine them with neural networks to enhance their performance remains to be explored. In this paper, we try to marry these two systems for text classification via structured language models. Specifically, we propose a novel label extraction framework based on binary syntax trees. Both the structures and intermediate representations of the trees are learned using a pretrained neural network in an unsupervised manner. Inference and learning is made efficient using dynamic programming over tree structures. Our experiments demonstrate that our approach could achieve good prediction results in single/multi-label text classification and have explicit and inherent constituent-level interpretability.

## 1 INTRODUCTION

Lack of interpretability is an intrinsic problem in deep neural networks based on layer-stacking for text classification. Many methods have been proposed to provide posthoc explanations for neural networks (Lipton, 2018; Lundberg & Lee, 2017; Sundararajan et al., 2017). However, these methods have multiple drawbacks. First, there is only word-level attribution but no high-level attribution such as those over phrases and clauses. Take sentiment analysis as an example, in addition to the ability to recognize the sentiment of sentences, an ideal interpretable model should be able to identify the sentiment and polarity reversal at the levels of words, phrases, and clauses. Secondly, as argued by Rudin (2019), models should be inherently interpretable rather than explained by a posthoc model.

A widely accepted property of natural languages is that *"the meaning of a whole is a function of the meanings of the parts and of the way they are syntactically combined"* (Partee, 1995). Compared with the sequential outputs of layer-stacked model architectures, syntactic tree structures naturally capture features of various levels because each node in a tree represents a constituent span. Such a characteristic motivates us to think about whether the representations of these internal nodes could be leveraged to design an inherently constituent-level interpretable model. One challenge faced by this idea is that traditional syntactic parsers require supervised training and have degraded performance on out-of-domain data. Fortunately, with the development of structured language models (Tu et al., 2013; Maillard et al., 2017; Choi et al., 2018; Kim et al., 2019), we are now able to learn hierarchical syntactic structures in an unsupervised manner from any raw text.

In this paper, we revisit interpretable text classification by using a structured language model as the backbone. We assume that each constituency in a text corresponds to at most one label and design a label extraction framework in accordance. Instead of treating constituents as separate objects and formulating text classification as multi-instance learning (Angelidis & Lapata, 2018), we take hierarchical constraints between spans into consideration. Specifically, given a tree whose nodes are associated with labels, we traverse the tree from its root in a top-down manner, then stop at nodes with task-defined labels, and gather all the labels. During training, we maximize the probability summation of all potential trees whose extracted labels are consistent with the gold label set. Though it is exponential to exhaust all potential labeled trees, we can estimate the sum of the probabilities of plausible trees through dynamic programming with linear complexity. As syntactic trees are

symbolic in nature, we derive neural constituent representations following the tree structures and hence build a Symbolic-Neural model.

The main contribution of this work is that we propose a Symbolic-Neural model, a simple but general model architecture for text classification, which has three advantages:

1. The model has both inherent interpretability and competitive prediction performance.
2. It handles both single-label and multi-label text classification tasks in a unified way instead of transferring the latter ones into binary classification problems (Read et al., 2011) like conventional methods.
3. It has not only word-level attribution but explicit multi-grained interpretability as well.

To the best of our knowledge, we are the first to introduce a text classification model with inherent constituent-level interpretability based on structured language models. We argue such characteristics of our model could be valuable in various application scenarios like data mining, NLU systems, prediction explanation, etc, and we discuss some of them in our experiments.

## 2 PRELIMINARY

### 2.1 ESSENTIAL PROPERTIES OF STRUCTURED LANGUAGE MODELS

Structured language models feature combining the powerful representation of neural networks with syntax structures. Though many attempts have been made about structured language models (Kim et al., 2019; Drozdov et al., 2019; Shen et al., 2021), three prerequisites need to be met before a model is selected as the backbone of our method. Firstly, it should have the ability to learn reasonable syntax structure in an unsupervised manner. Secondly, it computes an intermediate representation for each constituency node. Thirdly, it has a pretraining mechanism to improve representation performance. Since Fast-R2D2 (Hu et al., 2022) satisfies all the above conditions and also has good inference speed, we choose Fast-R2D2 as our backbone.

### 2.2 FAST-R2D2

Overall, Fast-R2D2 is a kind of structured language model that takes raw texts as inputs and outputs corresponding binary parsing trees along with node representations. The representation $e_{i,j}$ of each node $n_{i,j}$ is computed recursively from its child node representations via a shared composition function, i.e., $e_{i,j} = f(e_{i,k}, e_{k+1,j})$, where $k$ is the split point given by the parser and $f(\cdot)$ is an n-layered Transformer encoder. When $i = j$, $e_{i,j}$ is initialized as the embedding of the corresponding input token. Please note the parser is trained in an unsupervised manner, so no human-annotated parsing trees are required.

**Pretraining.** There are two components in Fast-R2D2: a top-down parser and a chart-based encoder R2D2 (Hu et al., 2021). For a given sentence $\mathbf{S}$, they denote the distribution of parser trees estimated by the parser and encoder as $q_\phi(\mathbf{z}|\mathbf{S})$ and $p_\theta(\mathbf{z}|\mathbf{S})$. They prune the chart by the top-down parser to achieve linear encoding complexity and optimize the parser by minimizing $D_{\mathrm{KL}}[q_\phi(\mathbf{z}|\mathbf{S}) \parallel p_\theta(\mathbf{z}|\mathbf{S})]$ denoted as $\mathcal{L}_{KL}$. Meanwhile, they pre-train the encoder by the bidirectional language model loss $\mathcal{L}_{bilm}$ proposed in R2D2. By minimizing $\mathcal{L}_{KL} + \mathcal{L}_{bilm}$, the parser and the encoder promote each other just like the strategy and value networks in AlphaZero (Silver et al., 2017).

## 3 METHODOLOGY

### 3.1 LABEL EXTRACTION FRAMEWORK

Unlike the conventional neural classification approach which inputs the whole sentence representation into an MLP layer, we predict the label of each node in the parse tree and output a final label set by the **yield** function introduced below.

**Inductive bias.** Through observing cases in single/multi-label classification tasks, we propose an inductive bias that a constituent in a text corresponds to at most one label. As constituents could be seen as nodes in a binary parsing tree, we can associate the nodes with labels. Nodes with multiple labels could be achieved by assigning labels to non-overlapping child nodes. Please note such an inductive bias is not applicable for special cases in which a minimal semantic constituent of a text is associated with multiple labels, e.g., the movie "Titanic" could be labeled with both 'disaster' and 'love'. However, we argue that such cases are rare because our inductive bias works well on most single/multi-label tasks as demonstrated in our experiments.

**Label Tree.** For a given sentence and its parsing tree, we associate each node with a label, which transfers the parsing tree to a label tree.

**Algorithm 1** Definition of Yield function

```
function YIELD(t̂)
    S = {}
    q ← [t̂.root]    ▷ The list of nodes to visit
    while len(q) > 0 do
        n_ij ← q.pop(0)
        if n_ij.label == φ_NT then
            if not n_ij.is_leaf then
                q.append(n_ij.left)
                q.append(n_ij.right)
        else
            S = S ∪ {n_ij.label}
    S = S \ {φ_T}
    return S
```

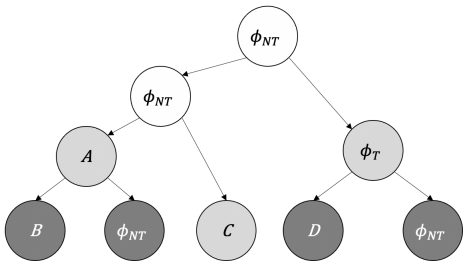



Figure 1: For the given tree, the **yield** function returns {A,C}. The yield function stops traversing at terminal nodes in shallow gray whose ancestor labels are all $\phi_{NT}$ and the nodes in dark gray are not visited.

**Yield function.** We design a **yield** function that traverses a label tree in a top-down manner and gathers task-defined labels of nodes. We divide the labels into two categories: terminal labels and non-terminal labels, which indicate whether the **yield** function should stop or continue respectively when it traverses to a node. Considering some nodes may not be associated with any task-defined labels, we introduce empty labels denoted as $\phi_T$ and $\phi_{NT}$ for terminal and non-terminal ones respectively. For simplicity, we don't discuss nesting cases in this paper, so there is only one unique non-terminal label which is $\phi_{NT}$ and all task-defined labels are terminal labels. However, our method can be naturally extended to handle nesting cases by allowing non-terminal labels to be associated with task labels. The **yield** function is defined by the pseudo-code in Algorithm 1. Figure 1 illustrates how the **yield** function traverses the label tree and gathers task-defined labels.

### 3.2 Symbolic-Neural Model

Given a sentence **S** and its best parsing tree $t$, we denote one of its potential label trees as $\hat{t}$. Note that $\hat{t}$ has the same tree structure as $t$. In addition, the best parsing tree is given by the unsupervised parser of Fast-R2D2 introduced in Sec 2.2. Let $\mathcal{Y}$ be the **yield** function. For a gold label set $\mathcal{T}$, the training objective is to maximize the probability summation of $\{\hat{t}|\mathcal{Y}(\hat{t}) = \mathcal{T}\}$. However, if we directly enumerate all potential label trees, we will encounter a combinatorial explosion. Fortunately, the probability summation could be estimated via dynamic programming.

Let $t_{i,j}$ denote the subtree spanning from $i$ to $j$ (both indices are inclusive), whose root, left and right children are $n_{i,j}$, $t_{i,k}$ and $t_{k+1,j}$ respectively in which $k$ is the split point. For each node

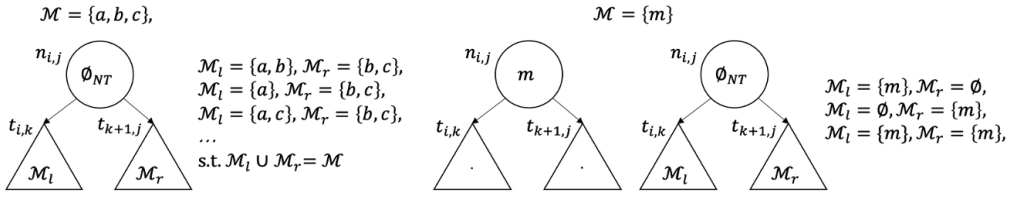



Figure 2: For $\mathcal{Y}(t_{i,j}) = \mathcal{M}$ subject to $|\mathcal{M}| >= 1$, the state spaces for dynamic programming.

$n_{i,j}$, its representation $e_{i,j}$ is computed by the composition function introduced in Sec 2.2, and the corresponding label probability distribution is $P(\cdot|n_{i,j}) = \mathrm{softmax}(\mathrm{MLP}(e_{i,j}))$ based on the inductive bias introduced in Sec 3.1. For a given tree $t_{i,j}$ and any label set $\mathcal{M}$, we denote the probability summation of $\{\hat{t}_{i,j}|\mathcal{Y}(\hat{t}_{i,j}) = \mathcal{M}\}$ as $P(\mathcal{M}|t_{i,j})$, abbreviated as $X_{i,j}^{\mathcal{M}}$. A naive way to estimate $X_{i,j}^{\mathcal{M}}$ is to enumerate all possible state spaces and sum them up via dynamic programming. Let $\mathcal{M}_l$ and $\mathcal{M}_r$ denote a pair of sets subject to $\mathcal{M}_l \cup \mathcal{M}_r = \mathcal{M}$. Let $\mathcal{C}(\mathcal{M})$ denote the set containing all valid $\mathcal{M}_l$ and $\mathcal{M}_r$ pairs. Figure 2 discusses all potential combinations of $\mathcal{M}_l$ and $\mathcal{M}_r$ when $|\mathcal{M}| >= 1$. For $\mathcal{M} = \phi$, $n_{i,j}$ could only be associated with $\phi_T$ or $\phi_{NT}$ with $\mathcal{M}_l = \phi$ and $\mathcal{M}_r = \phi$. Finally the transition function for $t_{i,j}$ where $i < j$ is:

$$X_{i,j}^{\mathcal{M}} = \begin{cases} P(\phi_{NT}|n_{i,j}) \cdot \sum_{\{\mathcal{M}_l,\mathcal{M}_r\}\in\mathcal{C}(\mathcal{M})} X_{i,k}^{\mathcal{M}_l} X_{k+1,j}^{\mathcal{M}_r} & ,|\mathcal{M}| > 1 \\ P(m|n_{i,j}) + P(\phi_{NT}|n_{i,j})(X_{i,k}^{\phi} X_{k+1,j}^{\mathcal{M}} + X_{i,k}^{\mathcal{M}} X_{k+1,j}^{\phi} + X_{i,k}^{\mathcal{M}} X_{k+1,j}^{\mathcal{M}}) & ,\mathcal{M} = \{m\} \\ P(\phi_T|n_{i,j}) + P(\phi_{NT}|n_{i,j})(X_{i,k}^{\phi} X_{k+1,j}^{\phi}) & ,\mathcal{M} = \phi \end{cases} \tag{1}$$

When $i = j$, we have:

$$X_{i,j}^{\mathcal{M}} = \begin{cases} 0 & ,|\mathcal{M}| > 1 \\ P(m|n_{i,j}) & ,\mathcal{M} = \{m\} \\ P(\phi_T|n_{i,j}) + P(\phi_{NT}|n_{i,j}) & ,\mathcal{M} = \phi \end{cases} \tag{2}$$

The transition function works in a bottom-up manner and iterates all possible $\mathcal{M} \subseteq \mathcal{T}$. $X_{1,n}^{\mathcal{T}}$ is the final probability summation of $\{\hat{t}_{1,n}|\mathcal{Y}(\hat{t}_{1,n}) = \mathcal{T}\}$. Even though, iterating $\mathcal{C}(\mathcal{M})$ and all $\mathcal{M} \subseteq \mathcal{T}$ is of exponential complexity, so it only works when $|\mathcal{T}|$ is small.

To tackle the problem of exponential complexity, we reformulate $P(\mathcal{T}|t)$ to $P(I^{\mathcal{T}} O^{\mathcal{F}\setminus\mathcal{T}}|t)$, where $I^{\mathcal{T}}$ and $O^{\mathcal{F}\setminus\mathcal{T}}$ denote the events of there being a $\hat{t}$ satisfying $\forall l \in \mathcal{T}, l \in \mathcal{Y}(\hat{t})$ and $\forall l \in \mathcal{F} \setminus \mathcal{T}, l \notin \mathcal{Y}(\hat{t})$ respectively. $\mathcal{F}$ denotes the full set of task labels. Let $\mathbb{I}(l)$ denote the event that there is a $\hat{t}$ satisfying $l \in \mathcal{Y}(\hat{t})$. By making conditional independence assumptions, we have:

$$\begin{aligned} &P(\mathcal{T}|t) = P(I^{\mathcal{T}} O^{\mathcal{F}\setminus\mathcal{T}}|t) \approx P(I^{\mathcal{T}}|t) \cdot P(O^{\mathcal{F}\setminus\mathcal{T}}|t) \\ &P(I^{\mathcal{T}}|t) \approx \Pi_{l\in\mathcal{T}} P(\mathbb{I}(l)|t), P(O^{\mathcal{F}\setminus\mathcal{T}}|t) = 1 - P(\cup_{l\in\mathcal{F}\setminus\mathcal{T}} \mathbb{I}(l)|t) \end{aligned} \tag{3}$$

We don't approximate $P(O^{\mathcal{F}\setminus\mathcal{T}}|t)$ as it could be computed directly.

For brevity, we denote $P(\mathbb{I}(l)|t_{i,j})$ as $Y_{i,j}^{l}$. Similarly to $\mathcal{M} = \{m\}$ in Figure 2, if there is a label tree $\hat{t}_{i,j}$ satisfying that $l \in \mathcal{Y}(\hat{t}_{i,j})$, the label of $n_{i,j}$ can only belong to $\{l, \phi_{NT}\}$. When $n_{i,j}$ is $\phi_{NT}$, it must satisfy $l \in \mathcal{Y}(\hat{t}_{i,k})$ or $l \in \mathcal{Y}(\hat{t}_{k+1,j})$. Therefore, we can establish the state transition equation of the nodes in $t$:

$$Y_{i,j}^{l} = \begin{cases} P(l|n_{i,j}) + P(\phi_{NT}|n_{i,j}) \cdot (1 - (1 - Y_{i,k}^{l}) \cdot (1 - Y_{k+1,j}^{l})) & \text{if } i < j \\ P(l|n_{i,j}) & \text{if } i = j \end{cases} \tag{4}$$

The above transition function premises that multiple non-overlapping spans could associate with the same label. In some cases, if there is a mutual-exclusiveness constraint that two non-overlapping spans aren't allowed to associate with the same task label, the transition function becomes:

$$Y_{i,j}^{l} = \begin{cases} P(l|n_{i,j}) + P(\phi_{NT}|n_{i,j}) \cdot (Y_{i,k}^{l} \cdot (1 - Y_{k+1,j}^{l}) + Y_{k+1,j}^{l} \cdot (1 - Y_{i,k}^{l})) & \text{if } i < j \\ P(l|n_{i,j}) & \text{if } i = j \end{cases} \tag{5}$$

Under such an assumption, if the label of the current node is not $l$, then only one of the left or right child nodes is allowed to associate with $l$.

As to $P(\cup_{l\in\mathcal{F}\setminus\mathcal{T}} \mathbb{I}(l)|t)$, we denote it as $Y_{i,j}^{\mathcal{F}\setminus\mathcal{T}}$ in short. Similar to Equation 4 we have:

$$Y_{i,j}^{\mathcal{F}\setminus\mathcal{T}} = \begin{cases} \sum_{l\in\mathcal{F}\setminus\mathcal{T}} P(l|n_{i,j}) + P(\phi_{NT}|n_{i,j}) \cdot (1 - (1 - Y_{i,k}^{\mathcal{F}\setminus\mathcal{T}}) \cdot (1 - Y_{k+1,j}^{\mathcal{F}\setminus\mathcal{T}})) & \text{if } i < j \\ \sum_{l\in\mathcal{F}\setminus\mathcal{T}} P(l|n_{i,j}) & \text{if } i = j \end{cases} \tag{6}$$

Thus $X_{1,n}^{\mathcal{T}} = \Pi_{l\in\mathcal{T}} Y_{1,n}^{l} \cdot (1 - Y_{1,n}^{\mathcal{F}\setminus\mathcal{T}})$ and the objective function given a parsing tree is:

$$\mathcal{L}_{cls}^{t} = -\log X_{1,n}^{\mathcal{T}} = -\sum_{l\in\mathcal{T}} \log Y_{1,n}^{l} - \log(1 - Y_{1,n}^{\mathcal{F}\setminus\mathcal{T}}) \tag{7}$$

Because it has been verified in the prior work (Hu et al., 2022) that models could achieve better downstream performance by training along with the parser objectives $\mathcal{L}_{bilm}$ and $\mathcal{L}_{KL}$ defined in Sec 2.2, we design the final loss as follows:

$$\mathcal{L} = \mathcal{L}^t_{cls} + \mathcal{L}_{bilm} + \mathcal{L}_{KL}\,, \quad t = \arg\max_z q_\phi(z|\mathbf{S}) \tag{8}$$

where $q_\phi(z|\mathbf{S})$ is the probability of a tree estimated by the parser in Fast-R2D2.

### 3.3 Top-down Encoder

As recursive language models follow a bottom-up hierarchical encoding process, context outside a span is invisible to the span, which may make low-level short spans unable to predict correct labels because of a lack of information. Here we draw on the ideas proposed by Le & Zuidema (2014); Drozdov et al. (2020) to allow information to flow in parse trees from top to down. The overall idea is to construct a top-down process to fuse information from both inside and outside of spans. For a given span $(i, j)$, we denote the top-down representation as $e'_{i,j}$. We use the Transformer as the top-down encoder function. The top-down encoding process starts from the root and functions recursively on the child nodes. For the root node, we have $[\cdot, e'_{1,n}] = f([e_{root}, e_{1,n}])$ where $e_{root}$ is embedding of the special token [ROOT]. Once the top-down representation $e'_{i,j}$ is ready, we compute its child representations recursively via $[\cdot, e'_{i,k}, e'_{k+1,j}] = f([e'_{i,j}, e_{i,k}, e_{k+1,j}])$ as illustrated in Figure 3. To enable the top-down encoder in the Symbolic-Neural model, we can simply replace $e_{i,j}$ with $e'_{i,j}$ in Sec 3.2.

## 4 Experiments

### 4.1 Downstream tasks

**Data set.** We report the results on the development set of the following datasets: SST-2, CoLA (Wang et al., 2019), ATIS (Hakkani-Tur et al., 2016), SNIPS (Coucke et al., 2018), StanfordLU (Eric et al., 2017). Please note that SST-2, CoLA, and SNIPS are single-label tasks and ATIS, StanfordLU are multi-label tasks. There are three sub-fields in StanfordLU including navigator, scheduler, and weather.

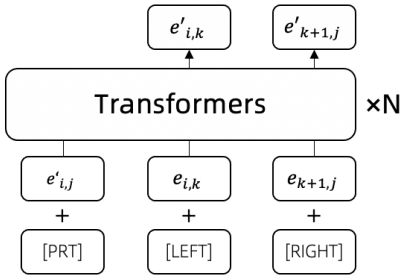



Figure 3: [PRT], [LEFT], [RIGHT] are role embeddings for the corresponding inputs.

**Baselines.** To fairly compare our method with other systems, all backbones such as Fast-R2D2 and BERT (Devlin et al., 2019) are pretrained on the same corpus with the same vocabulary and epochs. We record the best results of running with 4 different random seeds and report the mean of them. Because of GPU resource limit and energy saving, we pretrain all models on Wiki-103 (Merity et al., 2017), which contains 110 million tokens. To compare our model with systems only using whole sentence representations, we include BERT and Fast-R2D2 using root representation in our baselines. To study the reliability of the unsupervised parser, we include systems with a supervised parser Zhang et al. (2020) that uses BERT or a tree encoder as the backbone. For the former, we take the average pooling on representations of words in span (i,j) as the representation of the span. For the latter, we use the pretrained R2D2 tree encoder as the backbone. To compare with methods dealing with multi-instance learning (MIL) but without structure constraints, we extend the multi-instance learning framework proposed by Angelidis & Lapata (2018) to the multi-instance multi-label learning (MIMLL) scenario. Please find the details about the MIL and MIMLL in Appendix A.2. We also conduct ablation studies on systems with or without the top-down encoder and the mutual-exclusiveness constraint. For the systems using root or [CLS] representations on multi-label tasks, outputs are followed by a sigmoid layer and filtered by a threshold that is tuned on the training set.

**Hyperparameters.** Our BERT follows the setting in Devlin et al. (2019), using 12-layer Transformers with 768-dimensional embeddings, 3,072-dimensional hidden layer representations, and 12 attention heads. The setting of Fast-R2D2 follows Hu et al. (2022). Specifically, the tree encoder

uses 4-layer Transformers with other hyper-parameters same as BERT and the top-down encoder uses 2-layer ones. The top-down parser uses a 4-layer bidirectional LSTM with 128-dimensional embeddings and 256-dimensional hidden layers. We train all the systems across the seven datasets for 20 epochs with a learning rate of $5 \times 10^{-5}$ for the encoder, $1 \times 10^{-2}$ for the unsupervised parser, and batch size $64$ on 8 A100 GPUs.

| Backbone<br>Multi-Label% | Arch. | #Param. | SST-2<br>0 | CoLA<br>0 | SNIPS<br>0 | ATIS<br>1.69 | Nav.<br>26.54 | Sche.<br>24.86 | Wea.<br>5.39 |
|---|---|---|---|---|---|---|---|---|---|
| BERT(Wiki-103) | Sent. | 116M | 89.54 | 34.99 | 98.86 | 98.56 | 89.25 | **94.14** | 96.98 |
| parser+BERT | S.N. | 172M | 89.11 | 9.46 | 99.00 | 93.50 | 80.06 | 81.77 | 95.22 |
| parser+TreeEnc. | Sent. | 128M | 88.53 | 11.77 | 99.04 | 97.52 | 89.50 | 93.74 | 96.98 |
| parser+TreeEnc. | S.N. | 128M | 88.19 | 21.49 | 98.93 | 95.77 | 88,94 | 86.95 | 96.50 |
| Fast-R2D2 | MIL | 62M | 89.22 | 34.05 | 98.66 | - | - | - | - |
| Fast-R2D2 | MIMLL | 62M | - | - | - | 93.29 | 84.21 | 89.66 | 94.39 |
| Fast-R2D2 | Sent. | 62M | **89.96** | 36.31 | 99.00 | 98.11 | 89.25 | 93.68 | 96.70 |
| Fast-R2D2 | S.N.$_{fp}$ | 62M | - | - | - | 98.61 | 89.38 | 91.18 | 96.96 |
| Fast-R2D2$_{topdown}$ | S.N.$_{fp}$ | 67M | - | - | - | 98.45 | 88.91 | 93.46 | 97.13 |
| Fast-R2D2 | S.N. | 62M | 89.91 | 35.02 | 98.86 | **98.78** | 88.18 | 91.94 | **97.43** |
| Fast-R2D2$_{exclusive}$ | S.N. | 62M | 89.45 | **36.68** | 99.00 | 98.27 | 87.86 | 89.87 | 96.47 |
| Fast-R2D2$_{topdown}$ | S.N. | 67M | 89.45 | 36.49 | **99.14** | 98.50 | **90.82** | 93.47 | 97.21 |
| Fast-R2D2$_{top./excl.}$ | S.N. | 67M | 89.91 | 35.31 | 99.14 | 98.16 | 90.75 | 93.80 | 96.94 |

Table 1: We report mean accuracy for SST-2, Matthews correlation for CoLA, and F1 scores for the rest. We use 'S.N." to denote the systems based on the Symbolic-Neural architecture, and "Sent." to denote those using only whole sentence representations. We use subscript $fp$ for the models based on full permutation, $topdown$, and $exclusive$ for those with the top-down encoder and the mutual-exclusiveness constraint.

**Results and discussion.** We make several observations from Table 1. Firstly, We find that our models overall achieve competitive prediction accuracy compared with strong baselines including BERT, especially on multi-label tasks. The result validates the rationality of our label-constituent association inductive bias. The significant gap compared to MIMLL fully demonstrates the superiority of building hierarchical relationships between spans in the model. Secondly, when using sentence representation, the models with the unsupervised parser achieve similar results to those with the supervised parser on most tasks but significantly outperform the latter on CoLA. A possible reason for the poor performance of the latter systems on CoLA is that there are many sentences with grammar errors in the dataset which are not covered by the training set of the supervised parser. While the unsupervised parser can adapt to those sentences as $\mathcal{L}_{bilm}$ and $\mathcal{L}_{KL}$ are included in the final loss. The result reflects the flexibility and adaptability of using unsupervised parsers. Thirdly, 'parser+TreeEnc.' in Symbolic-Neural architectures does not perform as well as 'parser+TreeEnc.' using sentence representation, while the systems using the unsupervised parser show opposite results. Considering that the Symbolic-Neural model relies heavily on the representation of inner constituents, we suppose such results ascribe to the tree encoder having adapted to the trees given by the unsupervised parser during the pretraining stage of Fast-R2D2, which leads to the self-consistent intermediate representations. This result also verifies the structured language model that learns latent tree structures unsupervisedly is mature enough to be the backbone of our method.

### 4.2 Quantitative Analysis of Interpretability

To quantitatively compare the interpretability of different methods, we design a constituent-level attribution task to see how close the results learned by the models unsupervisedly are to human-annotated results. Specifically, we hide the gold span positions in NER and slot-filling datasets and make only raw text and gold span labels visible to models. We then treat these tasks as multi-label classification tasks, but we evaluate span positions learned unsupervisedly by models. Label trees of our model are generated by selecting the label with the max probability of each node.

**Data set.** We report F1 scores on the following data sets: ATIS (Hakkani-Tur et al., 2016), MITRestaurant (Liu et al., 2013a) and MITMovie (Liu et al., 2013b). ATIS is a slot-filling task and the others are NER tasks.

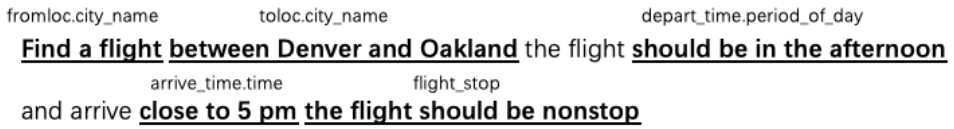



Figure 4: A sample of our method on semi-supervised slot filling. The ground truths are Denver, Oakland, afternoon, 5 pm, nonstop for each slot correspondingly. However, the last three are reasonable even though different from the ground truths.

**Baselines.** We include two baselines with interpretability on multi-label tasks: integrated-gradient (Sundararajan et al., 2017) and multi-instance learning (Angelidis & Lapata, 2018). We follow the setup in Sec 4.1 and report the results of the last epoch. For Integrated-Gradient, we use the same BERT in the last section as the encoder, filter the attribution of each token by a threshold and select filtered positions as outputs. For MIMLL, we select the span with the max attention score for a specified label. Please find details in Appendix A.2.

**Metrics.** We denote the predicted span set as $\mathcal{P}$ and gold span set as $\mathcal{G}$ and the overlap of $\mathcal{P}$ and $\mathcal{G}$ with the same labels as $\mathcal{O}$. Then we have:

$$\text{recall} = \frac{\sum_{o \in \mathcal{O}} o.j - o.i + 1}{\sum_{p \in \mathcal{P}} p.j - p.i + 1}, \text{prec} = \frac{\sum_{o \in \mathcal{O}} o.j - o.i + 1}{\sum_{g \in \mathcal{G}} g.j - g.i + 1}, \text{F1} = \frac{2 * (\text{prec} \cdot \text{recall})}{(\text{prec} + \text{recall})} \tag{9}$$

| Model | Thres. | Slot-filling | | | | sls-movie-eng | | | |
|---|---|---|---|---|---|---|---|---|---|
| length | | all | $1-2$ | $2-3$ | $3-5$ | all | $1-2$ | $2-5$ | $>5$ |
| ratio | | 100 | 95.84 | 3.70 | 0.46 | 100 | 55.60 | 42.75 | 1.65 |
| Integrated-Gradient$_{BERT}$ | 0.4 | **56.62** | **57.42** | **46.15** | 18.46 | 46.01 | 51.47 | 42.61 | 12.01 |
| Integrated-Gradient$_{BERT}$ | 0.5 | 56.24 | 57.44 | 39.09 | 13.79 | 42.46 | 49.79 | 37.42 | 13.06 |
| Integrated-Gradient$_{BERT}$ | 0.6 | 52.58 | 54.01 | 31.41 | 11.54 | 36.40 | 45.25 | 29.96 | 10.21 |
| MIMLL | N.A. | 11.11 | 10.84 | 17.37 | 16.74 | 14.55 | 14.11 | 14.76 | 17.43 |
| Symbolic-Neuron | N.A. | 35.30 | 35.38 | 33.78 | 34.01 | 53.04 | 50.61 | 54.99 | 57.77 |
| Symbolic-Neuron$_{exclusive}$ | N.A. | 32.13 | 32.37 | 30.62 | 16.33 | 52.89 | 50.45 | 54.55 | 61.15 |
| Symbolic-Neuron$_{topdown}$ | N.A. | 32.86 | 32.91 | 32.06 | 32.88 | 53.15 | 51.59 | 54.21 | 59.08 |
| Symbolic-Neuron$_{top./excl.}$ | N.A. | 42.01 | 42.28 | 38.69 | **34.95** | **57.82** | **56.54** | **58.87** | **59.82** |
| Model | Thres. | sls-movie-trivial | | | | sls-restaurant | | | |
| length | | all | $1-2$ | $2-5$ | $>5$ | all | $1-2$ | $2-5$ | $>5$ |
| ratio | | 100 | 7.57 | 57.07 | 35.36 | 100 | 40.87 | 57.89 | 1.24 |
| Integrated-Gradient$_{BERT}$ | 0.4 | 23.29 | 36.23 | 28.52 | 18.24 | 40.15 | 48.00 | 37.22 | 20.83 |
| Integrated-Gradient$_{BERT}$ | 0.5 | 18.53 | 34.75 | 24.23 | 13.07 | 36.21 | 45.11 | 32.76 | 17.56 |
| Integrated-Gradient$_{BERT}$ | 0.6 | 14.66 | 32.43 | 20.16 | 9.36 | 31.36 | 40.46 | 27.77 | 15.45 |
| MIMLL | N.A. | 61.77 | 42.48 | 52.03 | 69.02 | 7.58 | 7.40 | 7.73 | 6.08 |
| Symbolic-Neuron | N.A. | 67.30 | 41.11 | 60.18 | 75.18 | 48.07 | 42.93 | 50.89 | 45.60 |
| Symbolic-Neuron$_{exclusive}$ | N.A. | 63.60 | 44.75 | 58.89 | 68.80 | 49.46 | 44.28 | 52.22 | **48.20** |
| Symbolic-Neuron$_{topdown}$ | N.A. | 68.55 | 41.62 | 60.74 | 77.07 | 47.43 | 43.42 | 49.83 | 40.41 |
| Symbolic-Neuron$_{top./excl.}$ | N.A. | **70.83** | **45.92** | **64.32** | **77.73** | **52.52** | **49.14** | **54.67** | 44.27 |

Table 2: F1 scores for semi-supervised slot filling and NER whose golden span positions are hidden. "Thres." is short for threshold.

**Results and discussion.** From Table 2, one observation is that models with the mutual-exclusiveness constraint achieve better F1 scores. Such results illustrate that a stronger inductive bias is more helpful for models to learn constituent-label alignments. Besides, we find the Neural-Symbolic models significantly outperform the MIMLL and Integrated-Gradient baselines on the NER datasets but trail the Integrated-gradients on the slot-filling task. Through studying the outputs of our method, with a sample shown in Figure 4, we find that our model tends to recall long spans while the ground truths in ATIS tend to be short spans. We also find that on sls-movie-trivial, MIMLL significantly outperforms Integrated-Gradient. So we hypothesize that the distribution of

golden span lengths may affect results. We divide sentences into buckets according to the average golden span length and compute F1 scores for each bucket, as shown in Table 2. Interestingly, we find that the scores of Integrated-gradient decline significantly with increasing span lengths, while our method performs well on all the buckets. In addition, we argue that the F1 scores on the NER datasets can reflect interpretability more objectively, because the boundaries of proper nouns are clear and objective, while the choice of slots is relatively ambiguous about whether to include prepositions, modal verbs, etc.

### 4.3 Case Study & Potential Applications

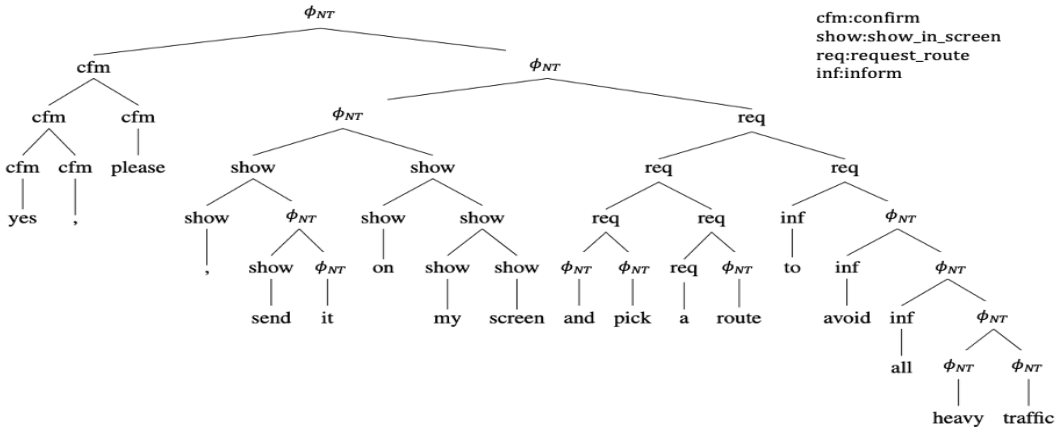



Figure 5: A sample of the symbolic-neural model on Navigator with the top-down encoder.

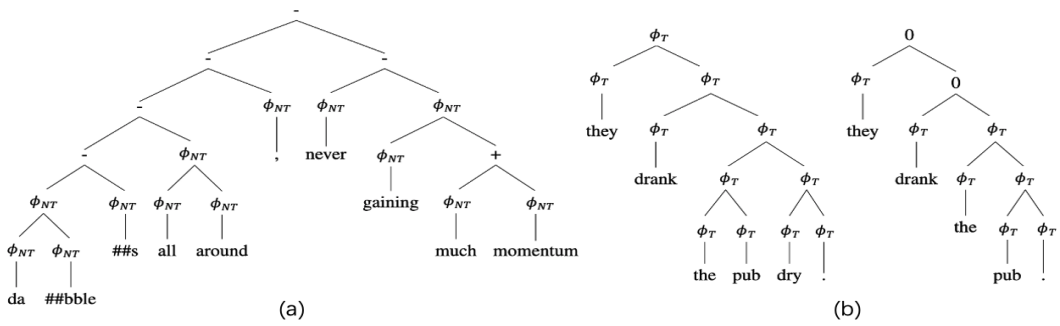



Figure 6: Samples of the symbolic-neural model with the top-down encoder.

We output the label trees generated by our model trained on the Navigator, SST-2, and CoLA to observe whether the model has sufficient interpretability. From Figure 5 we can find our method is able to learn potential alignments of intents and texts and show them explicitly. This can be used in multi-intent NLU systems to help determine the attribution of slots to corresponding intents. We also study the difference between generated label trees of the vanilla Symbolic-Neural model and the Symbolic-Neural$_{topdown}$. The cases could be found in Appendix A.4. We find the vanilla Symbolic-Neural model fails to deal with multi-intent cases. Such an observation verifies the necessity of introducing the top-down encoder. For SST-2, as there are no neutral samples, we randomly sampled sentences from Wiki-103 as neutral texts and force all nodes to be $\phi_{NT}$ by the mean squared error loss. Figure 6(a) shows the sentiment polarity of each constituent and the polarity reversal of "never". Such a characteristic could be used for text mining by gathering the minimal spans of a specified label. We also study the generated label trees on CoLA, a linguistic acceptance data set. We transfer the task to a grammar error detection problem by converting the label "1" to $\phi$ as "1" means no error is found in a sentence. Figure 6(b) shows it's able to detect incomplete constituents and may help in applications like grammar error location. More cases could be found in the Appendix.

## 5 RELATED WORKS

**Structured language models.** Many attempts have been made to develop structured language models. Pollack (1990) proposed to use RvNN as a recursive architecture to encode text hierarchically, and Socher et al. (2013) showed the effectiveness of RvNNs with gold trees for sentiment analysis. However, both approaches require annotated trees. Gumbel-Tree-LSTMs (Choi et al., 2018) construct trees by recursively selecting two terminal nodes to merge and learning composition probabilities via downstream tasks. CRvNN (Chowdhury & Caragea, 2021) makes the entire process end-to-end differentiable and parallel by introducing a continuous relaxation. However, neither Gumbel-Tree-LSTMs nor CRvNN mentions the pretraining mechanism in their work. URNNG (Kim et al., 2019) proposed the first architecture to jointly pretrain a parser and an encoder based on RNNG (Dyer et al., 2016). However, its $O(n^3)$ time and space complexity makes it hard to pretrain on large-scale corpora. ON-LSTM and StructFormer (Shen et al., 2019; 2021) propose a series of methods to integrate structures into LSTM or Transformer by masking information in differentiable ways. As the encoding process is still performed in layer-stacking models, there are no intermediate representations for tree nodes. Maillard et al. (2017) propose an alternative approach, based on a differentiable CKY encoding. The algorithm is differentiable by using a soft-gating approach, which approximates discrete candidate selection by a probabilistic mixture of the constituents available in a given cell of the chart. While their work relies on annotated downstream tasks to learn structures,Drozdov et al. (2019) propose a novel auto-encoder-like pretraining objective based on the inside-outside algorithm Baker (1979); Casacuberta (1994) but is still of cubic complexity. To tackle the $O(n^3)$ limitation of CKY encoding, Hu et al. (2021) propose an MLM-like pretraining objective and a pruning strategy, which reduces the complexity of encoding to linear and makes the model possible to pretrain on large-scale corpora.

**Multi-Instance Learning.** Multi-Instance learning (MIL) deals with problems where labels are associated with groups of instances or bags (spans in our case), while instance labels are unobserved. The goal is either to label bags Keeler et al. (1990); Dietterich et al. (1997); Maron & Ratan (1998) or to simultaneously infer bag and instance labels Zhou et al. (2009); Kotzias et al. (2015). Angelidis & Lapata (2018) apply MIL to segment-level sentiment analysis based on an attention-based scoring method. In our work, we refine instances to different semantic granularities and consider hierarchical relationships between instances.

**Model Interpretability.** In the line of work on model interpretability, many approaches have been proposed. Ribeiro et al. (2016); Lundberg & Lee (2017) try to generate explanation for prediction. Baehrens et al. (2010); Simonyan et al. (2014); Sundararajan et al. (2017) analyze attribution by gradients. The above-mentioned methods are all posthoc. Kim et al. (2020); Cao et al. (2020); Chen & Ji (2020) apply masks on the model input in text classification to obtain token weights, but single-dimensional weights are not enough to reflect multi-label interpretation. Alvarez-Melis & Jaakkola (2018); Rudin (2019) argue interpretability should be an inherent property of a deep neural network and propose corresponding model architectures. However, all the above-mentioned methods are not able to generate constituent-level interpretability.

## 6 CONCLUSION, LIMITATION AND FUTURE WORK

In this paper, we propose a novel label extraction framework based on a simple inductive bias and model single/multi-label text classification in a unified way. We discuss how to build a probabilistic model to maximize the valid potential label trees by leveraging the internal representations of a structured language model as symbolic interfaces. Our experiment results show our method achieves inherent interpretability on various granularities. The generated label trees could have potential values in various unsupervised tasks requiring constituent-level outputs.

Regarding to the limitation of our work, we require that the labels corresponding to the texts in the dataset have a certain degree of diversity, thus forcing the model to learn self-consistent constituent-label alignments. For example, in ATIS, almost all training samples have the same labels like "from-loc.city_name" and "toloc.city_name". That's why our model fails to accurately associate these two labels with correct spans in Figure 4.

In the future, we will explore more unsupervised application scenarios, such as text-mining combined with VAE (Kingma & Welling, 2014) and image-text alignment tasks, to make the architecture more versatile.

## 7 REPRODUCIBILITY STATEMENT

In the supplemental, we include a zip file containing our code and datasets downloading linkage. We've also included in the supplemental the scripts we run all baselines and the Symbolic-Neural models.

## REFERENCES

David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pp. 7786–7795, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/3e9f0fc9b2f89e043bc6233994dfcf76-Abstract.html.

Stefanos Angelidis and Mirella Lapata. Multiple instance learning networks for fine-grained sentiment analysis. *Trans. Assoc. Comput. Linguistics*, 6:17–31, 2018. doi: 10.1162/tacl_a_00002. URL https://doi.org/10.1162/tacl_a_00002.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *J. Mach. Learn. Res.*, 11: 1803–1831, 2010. doi: 10.5555/1756006.1859912. URL https://dl.acm.org/doi/10.5555/1756006.1859912.

James K. Baker. Trainable grammars for speech recognition. *Journal of the Acoustical Society of America*, 65, 1979.

Nicola De Cao, Michael Sejr Schlichtkrull, Wilker Aziz, and Ivan Titov. How do decisions emerge across layers in neural models? interpretation with differentiable masking. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 3243–3255. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.262. URL https://doi.org/10.18653/v1/2020.emnlp-main.262.

Francisco Casacuberta. Statistical estimation of stochastic context-free grammars using the inside-outside algorithm and a transformation on grammars. In Rafael C. Carrasco and José Oncina (eds.), *Grammatical Inference and Applications, Second International Colloquium, ICGI-94, Alicante, Spain, September 21-23, 1994, Proceedings*, volume 862 of *Lecture Notes in Computer Science*, pp. 119–129. Springer, 1994. doi: 10.1007/3-540-58473-0_142. URL https://doi.org/10.1007/3-540-58473-0_142.

Hanjie Chen and Yangfeng Ji. Learning variational word masks to improve the interpretability of neural text classifiers. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 4236–4251. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.347. URL https://doi.org/10.18653/v1/2020.emnlp-main.347.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. Learning to compose task-specific tree structures. In Sheila A. McIlraith and Kilian Q. Weinberger (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 5094–5101. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16682.

Jishnu Ray Chowdhury and Cornelia Caragea. Modeling hierarchical structures with continuous recursive neural networks. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 1975–1988. PMLR, 2021. URL http://proceedings.mlr.press/v139/chowdhury21a.html.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *CoRR*, abs/1805.10190, 2018. URL http://arxiv.org/abs/1805.10190.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL https://doi.org/10.18653/v1/n19-1423.

Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1-2):31–71, 1997. doi: 10.1016/S0004-3702(96)00034-3. URL https://doi.org/10.1016/S0004-3702(96)00034-3.

Andrew Drozdov, Patrick Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. Unsupervised latent tree induction with deep inside-outside recursive auto-encoders. 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)*, pp. 1129–1141, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1116. URL https://www.aclweb.org/anthology/N19-1116.

Andrew Drozdov, Subendhu Rongali, Yi-Pei Chen, Tim O'Gorman, Mohit Iyyer, and Andrew McCallum. Unsupervised parsing with S-DIORA: Single tree encoding for deep inside-outside recursive autoencoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4832–4845, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.392. URL https://www.aclweb.org/anthology/2020.emnlp-main.392.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 199–209, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1024. URL https://www.aclweb.org/anthology/N16-1024.

Mihail Eric, Lakshmi Krishnan, François Charette, and Christopher D. Manning. Key-value retrieval networks for task-oriented dialogue. In Kristiina Jokinen, Manfred Stede, David DeVault, and Annie Louis (eds.), *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, pp. 37–49. Association for Computational Linguistics, 2017. doi: 10.18653/v1/w17-5506. URL https://doi.org/10.18653/v1/w17-5506.

Dilek Hakkani-Tur, Gokhan Tur, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Proceedings of Interspeech*, 2016.

Xiang Hu, Haitao Mi, Zujie Wen, Yafang Wang, Yi Su, Jing Zheng, and Gerard de Melo. R2D2: Recursive transformer based on differentiable tree for interpretable hierarchical language modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4897–4908, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.379. URL https://aclanthology.org/2021.acl-long.379.

Xiang Hu, Haitao Mi, Liang Li, and Gerard de Melo. Fast-r2d2: A pretrained recursive neural network based on pruned cky for grammar induction and text representation, 2022. URL https://arxiv.org/abs/2203.00281.

James D. Keeler, David E. Rumelhart, and Wee Kheng Leow. Integrated segmentation and recognition of hand-printed numerals. In Richard Lippmann, John E.

Moody, and David S. Touretzky (eds.), *Advances in Neural Information Processing Systems 3, [NIPS Conference, Denver, Colorado, USA, November 26-29, 1990]*, pp. 557–563. Morgan Kaufmann, 1990. URL http://papers.nips.cc/paper/397-integrated-segmentation-and-recognition-of-hand-printed-numerals.

Siwon Kim, Jihun Yi, Eunji Kim, and Sungroh Yoon. Interpretation of NLP models through input marginalization. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pp. 3154–3167. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.255. URL https://doi.org/10.18653/v1/2020.emnlp-main.255.

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. Unsupervised recurrent neural network grammars. 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)*, pp. 1105–1117, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1114. URL https://www.aclweb.org/anthology/N19-1114.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Dimitrios Kotzias, Misha Denil, Nando de Freitas, and Padhraic Smyth. From group to individual labels using deep features. In Longbing Cao, Chengqi Zhang, Thorsten Joachims, Geoffrey I. Webb, Dragos D. Margineantu, and Graham Williams (eds.), *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pp. 597–606. ACM, 2015. doi: 10.1145/2783258.2783380. URL https://doi.org/10.1145/2783258.2783380.

Phong Le and Willem Zuidema. Inside-outside semantics: A framework for neural models of semantic composition. In *NIPS 2014 Workshop on Deep Learning and Representation Learning*, 2014.

Zachary C. Lipton. The mythos of model interpretability. *Commun. ACM*, 61(10):36–43, 2018. doi: 10.1145/3233231. URL https://doi.org/10.1145/3233231.

Jingjing Liu, Panupong Pasupat, Scott Cyphers, and James R. Glass. Asgard: A portable architecture for multilingual dialogue systems. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 8386–8390. IEEE, 2013a. doi: 10.1109/ICASSP.2013.6639301. URL https://doi.org/10.1109/ICASSP.2013.6639301.

Jingjing Liu, Panupong Pasupat, Yining Wang, Scott Cyphers, and James R. Glass. Query understanding enhanced by hierarchical parsing structures. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding, Olomouc, Czech Republic, December 8-12, 2013*, pp. 72–77. IEEE, 2013b. doi: 10.1109/ASRU.2013.6707708. URL https://doi.org/10.1109/ASRU.2013.6707708.

Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 4765–4774, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

Jean Maillard, Stephen Clark, and Dani Yogatama. Jointly learning sentence embeddings and syntax with unsupervised tree-lstms. *CoRR*, abs/1705.09189, 2017. URL http://arxiv.org/abs/1705.09189.

Oded Maron and Aparna Lakshmi Ratan. Multiple-instance learning for natural scene classification. In Jude W. Shavlik (ed.), *Proceedings of the Fifteenth International Conference on Machine Learning (ICML 1998), Madison, Wisconsin, USA, July 24-27, 1998*, pp. 341–349. Morgan Kaufmann, 1998.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Barbara H. Partee. Lexical semantics and compositionality. 1995.

Jordan B. Pollack. Recursive distributed representations. *Artif. Intell.*, 46(1-2):77–105, 1990. doi: 10.1016/0004-3702(90)90005-K. URL https://doi.org/10.1016/0004-3702(90)90005-K.

Jesse Read, Bernhard Pfahringer, Geoff Holmes, and Eibe Frank. Classifier chains for multi-label classification. *Mach. Learn.*, 85(3):333–359, 2011. doi: 10.1007/s10994-011-5256-5. URL https://doi.org/10.1007/s10994-011-5256-5.

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 97–101, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-3020. URL https://aclanthology.org/N16-3020.

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

Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron C. Courville. Ordered neurons: Integrating tree structures into recurrent neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=B1l6qiR5F7.

Yikang Shen, Yi Tay, Che Zheng, Dara Bahri, Donald Metzler, and Aaron C. Courville. Structformer: Joint unsupervised induction of dependency and constituency structure from masked language modeling. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pp. 7196–7209. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.559. URL https://doi.org/10.18653/v1/2021.acl-long.559.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy P. Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nat.*, 550(7676):354–359, 2017. doi: 10.1038/nature24270. URL https://doi.org/10.1038/nature24270.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6034.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1631–1642. ACL, 2013. URL https://aclanthology.org/D13-1170/.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328. PMLR, 2017. URL http://proceedings.mlr.press/v70/sundararajan17a.html.

Kewei Tu, Maria Pavlovskaia, and Song Chun Zhu. Unsupervised structure learning of stochastic and-or grammars. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pp. 1322–1330, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/24681928425f5a9133504de568f5f6df-Abstract.html.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJ4km2R5t7.

Yu Zhang, Houquan Zhou, and Zhenghua Li. Fast and accurate neural CRF constituency parsing. In *Proceedings of IJCAI*, pp. 4046–4053, 2020. doi: 10.24963/ijcai.2020/560. URL https://doi.org/10.24963/ijcai.2020/560.

Zhi-Hua Zhou, Yu-Yin Sun, and Yu-Feng Li. Multi-instance learning by treating instances as non-i.i.d. samples. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman (eds.), *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pp. 1249–1256. ACM, 2009. doi: 10.1145/1553374.1553534. URL https://doi.org/10.1145/1553374.1553534.

# A Appendix

## A.1 Multi-Label Learning based on Fast-R2D2

We adopt a canonical multi-instance learning framework used in text classification proposed by Angelidis & Lapata (2018), in which each instance has a representation and all instances are fused by attention. The original work produce hidden vectors $h_i$ for each segment by GRU modules and compute attention weights $a_i$ as the normalized similarity of each $h_i$ with $h_a$.

$$a_i = \frac{exp(\mathrm{h}_i^{\mathsf{T}}\mathrm{h}_a)}{\sum_i exp(\mathrm{h}_i^{\mathsf{T}}\mathrm{h}_a)}\,, p_i = \mathrm{softmax}(W_{cls}h_i + b_{cls})\,, p_d^{(c)} = \sum_i a_i p_i^{(c)}\,,\ c \in [1, C]\,. \tag{10}$$

where $C$ is the total class number, $p_i$ is the individual segment label prediction, $p_d$ is document level predictions. They use the negative log-likelihood of the prediction as an objective function: $L_{cls} = -\sum_d \log p_d^{(y_d)}$. We simply replace segment representations with span representations in our work as the experiment baseline. Specifically, we use the top-down representation $e'_{i,j}$ as the tensor to be attended to and predict the label by $e_{i,j}$:

$$\begin{aligned} a_{i,j} &= \frac{exp(\mathrm{e}'^{\mathsf{T}}_{\mathrm{i,j}}\mathrm{h}_a)}{\sum_{m,n\in\mathcal{D}} exp(\mathrm{e}'^{\mathsf{T}}_{\mathrm{m,n}_i}\mathrm{h}_a)}\,, p_{i,j} = \mathrm{softmax}(W_{cls}e_{i,j} + b_{cls})\,, \\ p_d^{(c)} &= \sum_{m,n\in\mathcal{D}} a_{m,n}p_{m,n}^{(c)}\,,\ c \in [1, C]\,. \end{aligned} \tag{11}$$

where $\mathcal{D}$ is the span set for a parsing tree. Please note the MIL model in our baselines is trained together with $\mathcal{L}_{bilm}$ and $\mathcal{L}_{KL}$, whose final loss is $\mathcal{L}_{cls} + \mathcal{L}_{bilm} + \mathcal{L}_{KL}$.

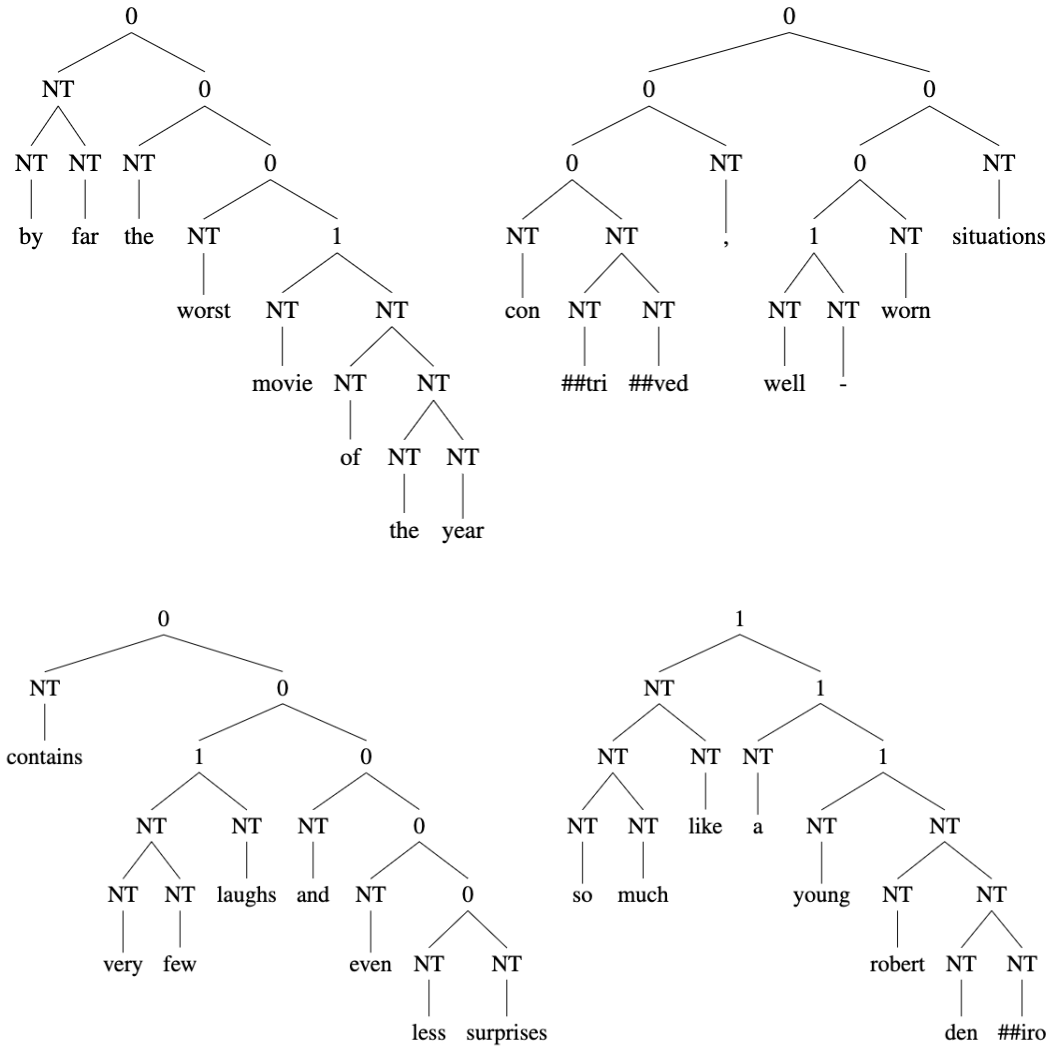
### A.2 MULTI-LABEL MULTI-INSTANCE LEARNING BASED ON FAST-R2D2

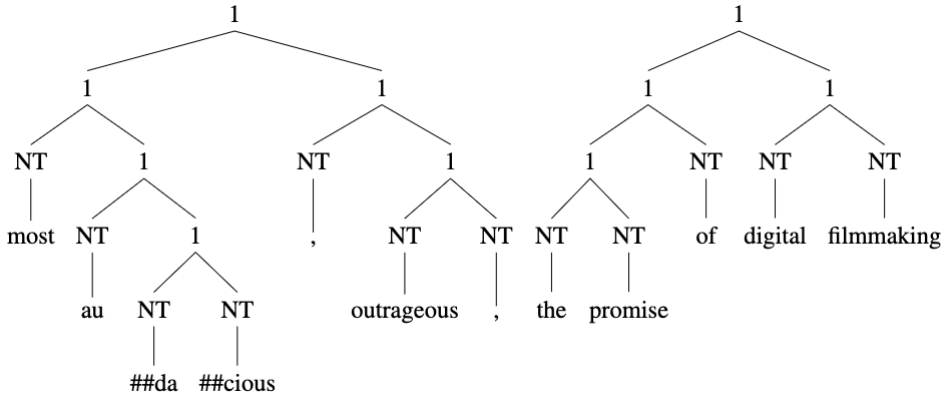
To support multi-label multi-instance learning, we refactor the above equations to enable them to support attention on different labels. For each label there is vector $h_a^{(c)}$, based on which attention weights $a_i^{(c)}$ is computed as following:

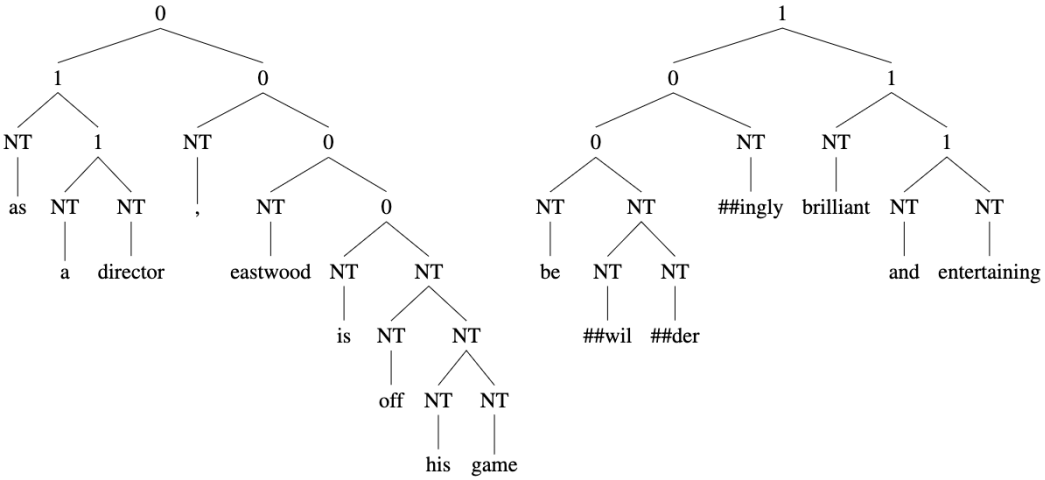
$$
\begin{aligned}
a_{i,j}^{(c)} &= \frac{exp(\mathrm{e}'^{\mathsf{T}}_{i,j}\mathrm{h}_a^{(c)})}{\sum_{m,n\in\mathcal{D}} exp(\mathrm{e}^{\mathsf{T}}_{m,n}\mathrm{h}_a^{(c)})}\,, p_{i,j}^{(c)} = sigmoid(W_{cls}^{(c)}e_{i,j} + b_{cls}^{(c)})\,, \\
p^{(c)} &= \sum_{m,n\in\mathcal{D}} a_{m,n}p_{m,n}^{(c)}\,,\ c \in [1, C]\,.
\end{aligned}
\tag{12}
$$

The final objective function is $L = -\sum_{c\in\mathcal{T}} \log p^{(c)} - \sum_{c\in\mathcal{F}\setminus\mathcal{T}} \log(1-p^{(c)})$. In the semi-supervised slot-filling and NER tasks, we let the model predicts labels first and then pick the span with the max attention weight for each label.

### A.3 REAL LABEL TREES SAMPLED FROM SYMBOLIC-NEURAL$_{-t/-e}$ ON SST-2




1
1
1
NT
1
NT
1
most
NT
1
,
NT
NT
au
NT
NT
outrageous
,
##da
##cious
1
1
1
1
NT
NT
NT
NT
NT
of
digital
filmmaking
the
promise


0
1
0
NT
1
NT
0
as
NT
NT
,
NT
0
a
director
eastwood
NT
NT
is
NT
NT
off
NT
NT
his
game
1
0
1
0
NT
NT
##ingly
brilliant
NT
NT
NT
1
be
NT
NT
and
entertaining
##wil
##der

### A.4 SAMPLED LABEL TREES IN ATIS

We sample label trees from Neural-Symbolic$_{-t/-e}$ and Neural-Symbolic$_{+t/-e}$ respectively for observation. Ground truths are annotated in brackets.

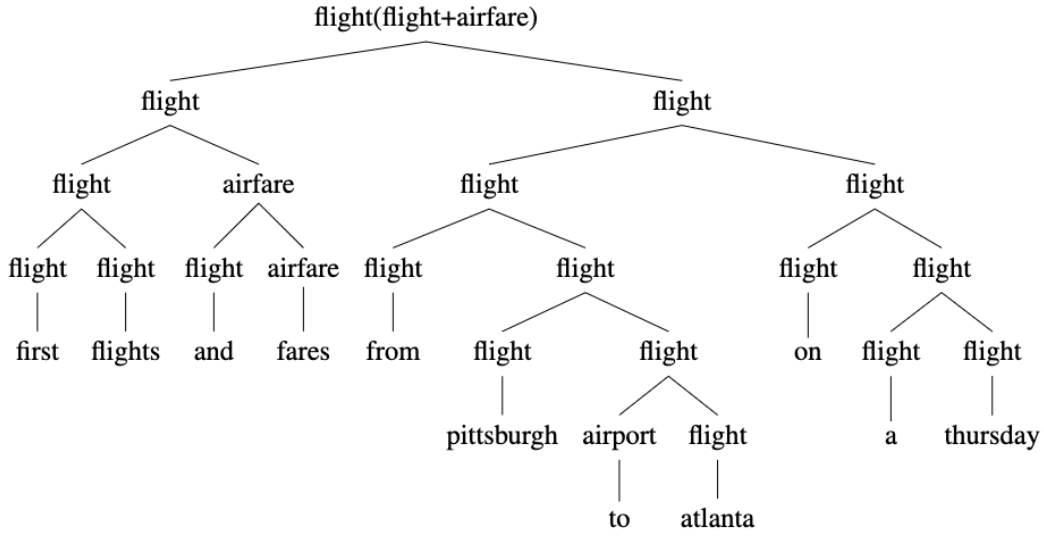



Figure 7: The label tree generated by Neural-Symbolic w/o the topdown encoder.

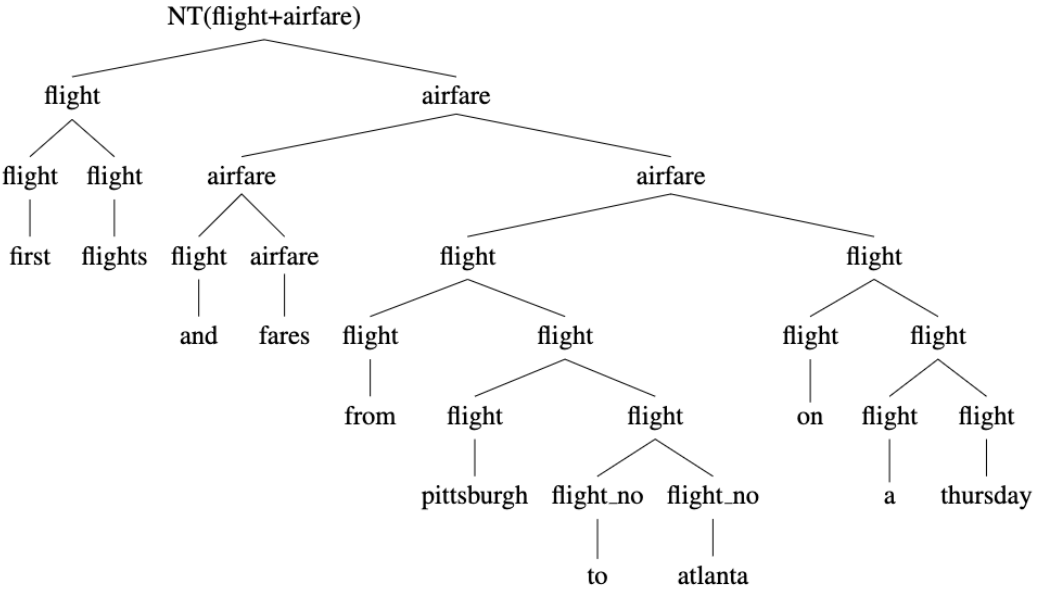



Figure 8: The label tree generated by Neural-Symbolic$_{topdown}$

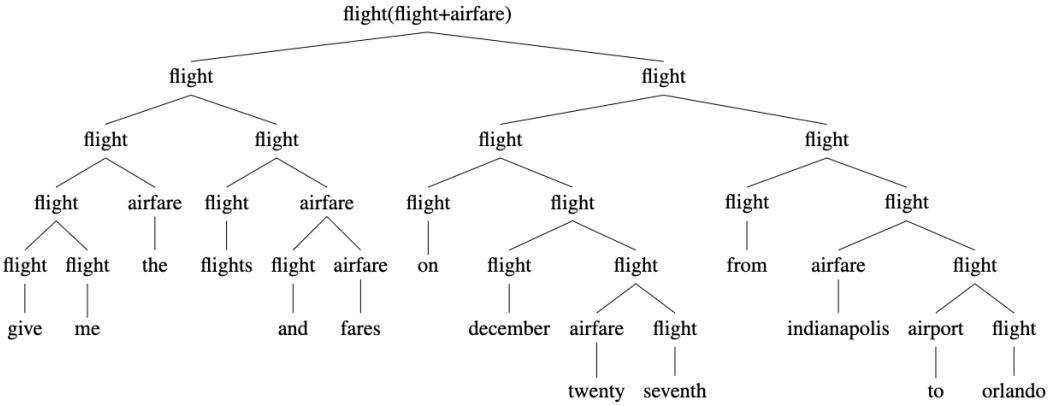



Figure 9: The label tree generated by Neural-Symbolic

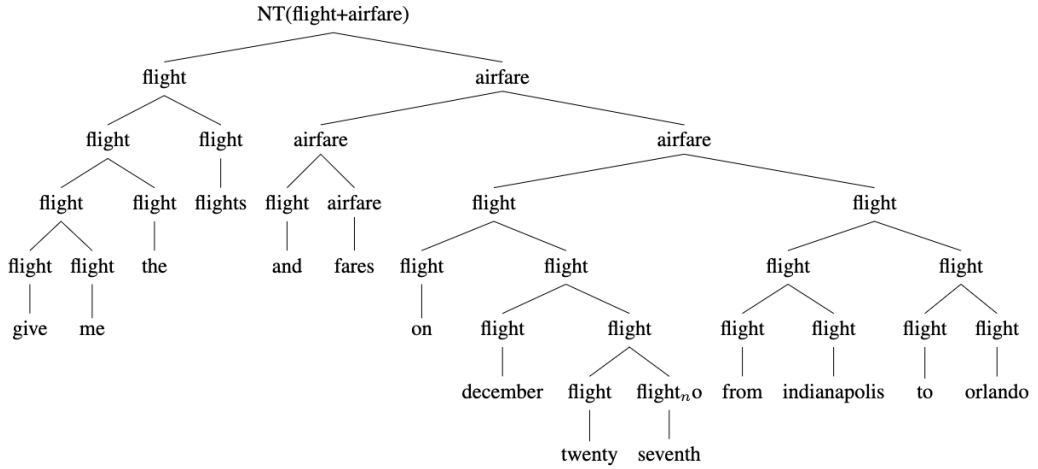



Figure 10: The label tree generated by Neural-Symbolic$_{topdown}$

### A.5 SAMPLED LABEL TREES IN NAVIGATOR

2:request_route, 3:appreciate, 4:request_address, 6:navigate

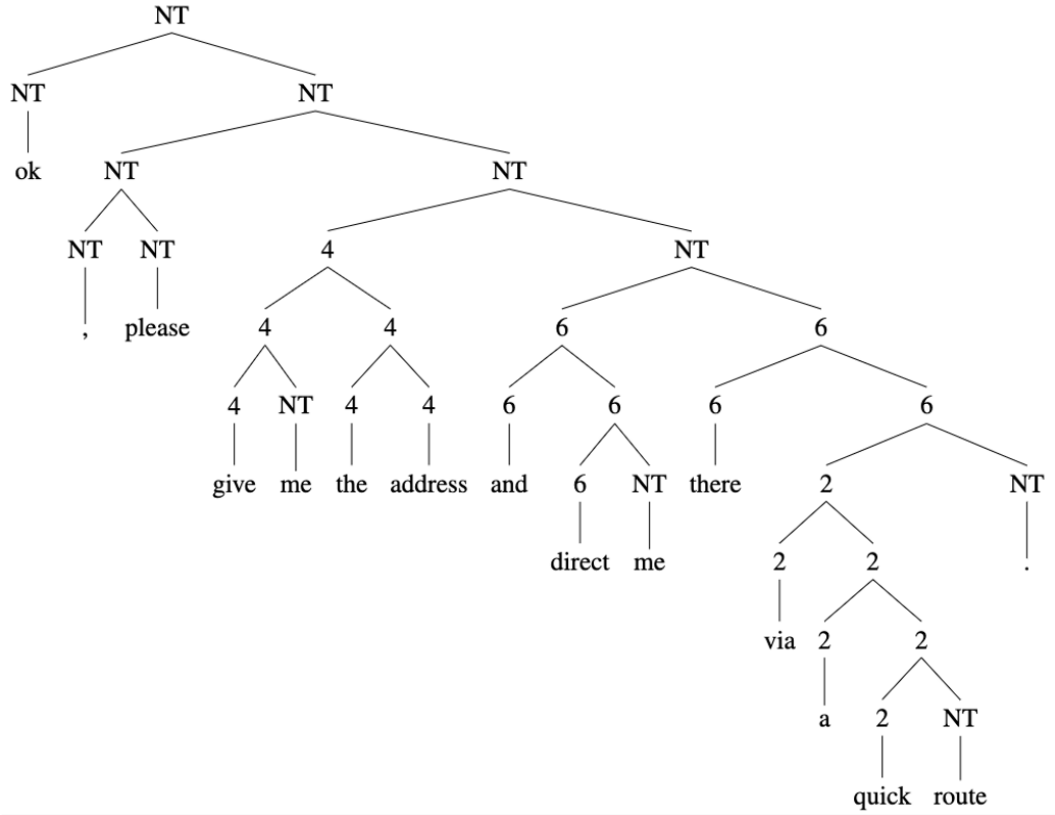



Figure 11: The label tree generated by Neural-Symbolic$_{topdown}$

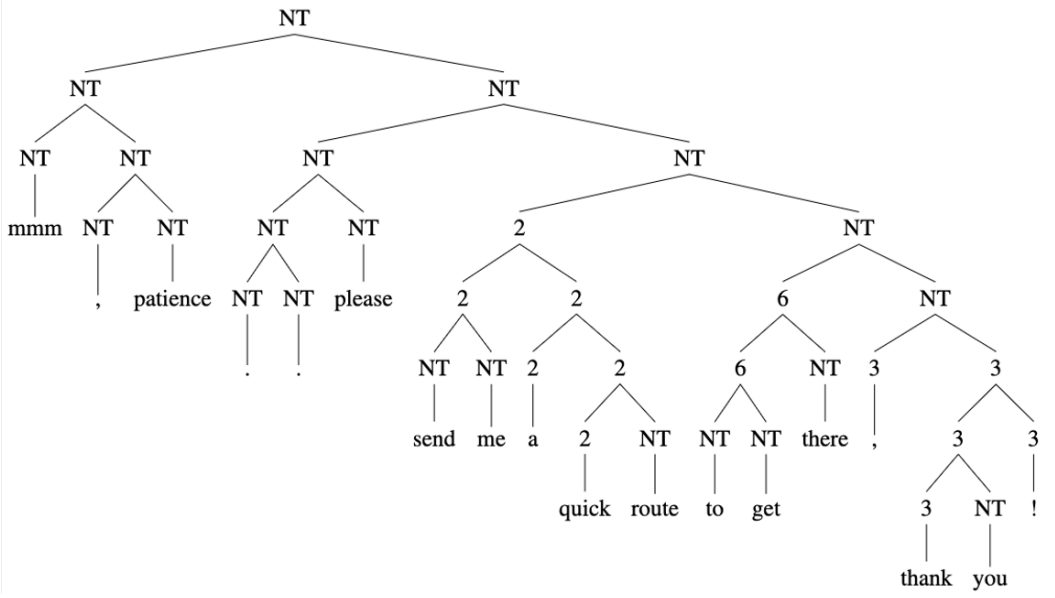



Figure 12: The label tree generated by Neural-Symbolic$_{topdown}$