Robust Frame-Semantic Models with Lexical Unit Trees and Negative Samples

Jacob Devasier **Yogesh Gurjar** **Chengkai Li**
The University of Texas at Arlington
jacob.devasier@mavs.uta.edu
yxg7510@mavs.uta.edu
cli@uta.edu

Abstract

We present novel advancements in frame-semantic parsing, specifically focusing on target identification and frame identification. Our target identification model employs a novel prefix tree modification to enable robust support for multi-word lexical units, resulting in a coverage of 99.4% of the targets in the FrameNet 1.7 fulltext annotations. It utilizes a RoBERTa-based filter to achieve an F1 score of 0.775, surpassing the previous state-of-the-art solution by +0.012. For frame identification, we introduce a modification to the standard multiple-choice classification paradigm by incorporating additional negative frames for targets with limited candidate frames, resulting in a +0.014 accuracy improvement over the frame-only model of FIDO, the previous state-of-the-art system, and +0.002 over its full system. Our approach significantly enhances performance on rare frames, exhibiting an improvement of +0.044 over FIDO's accuracy on frames with 5 or fewer samples, and on under-utilized frames, with an improvement of +0.139 on targets with a single candidate frame. Overall, our contributions address critical challenges and advance the state-of-the-art in frame-semantic parsing.

1 Introduction

Frame-semantic parsing (Gildea and Jurafsky, 2002) is the process of automatically identifying semantic frames (Fillmore and Baker, 2009) and frame elements within text. Semantic frames are structured events, concepts, or scenarios containing frame elements which describe different roles or entities associated with those events, concepts, or scenarios. Semantic frames provide a structured framework for performing and explaining natural language processing tasks, such as knowledge extraction (Søgaard et al., 2015; Si and Roberts, 2018), question answering (Gildea and Jurafsky, 2002), fact checking (Arslan et al., 2020), and event detection (Spiliopoulou et al., 2017). An example

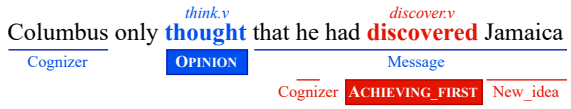

Figure 1: An example of frame-semantic parsing annotations in FrameNet (Baker et al., 1998). Targets are represented by bold words with their corresponding lexical unit italicized above and evoked frame boxed below. Frame elements are labeled below the text and are color-coded to match their corresponding frame.

of frame-semantic parsing annotations for a sentence is shown in Figure 1.

Frame-semantic parsing typically consists of three primary tasks: *target identification*, *frame identification*, and *argument identification* (Das et al., 2014). Target identification is the identification of targets—words or phrases that evoke frames in a sentence, e.g., *thought* and *discovered* in Figure 1. Each target is an instance of a particular lexical unit, a linguistic unit which pairs a word with a meaning, represented as *lemma.part-of-speech*, e.g., *think.v* for *thought* and *discover.v* for *discovered* in Figure 1. Frame identification is the classification of each target as a frame, e.g., OPINION and ACHIEVING_FIRST in Figure 1. Argument identification is the identification and classification of each frame element belonging to a frame evoked by a target, e.g., "Columbus", the Cognizer frame element, and "that he [...] Jamaica", the Message frame element, in the OPINION frame in Figure 1.

In this paper, we present novel methods for target identification and frame identification which outperform previous state-of-the-art approaches. Our target identification model generates a robust set of candidate targets for each lexical unit in the FrameNet 1.7 fulltext annotations (Baker et al., 1998), covering 99.4% of the targets in the test set, before filtering out false-positive candidates by employing a RoBERTa (Liu et al., 2019) classification model.

Previous works (Johansson and Nugues, 2007;

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6930–6941

Das et al., 2014) have also approached target identification using a candidate generation and filtering process due to poor performance of statistical models on directly predicting targets. However, neither of these were able to identify discontinuous lexical units– multi-word lexical units separated by auxiliary words. We address this problem by representing candidate lexical units in a modified prefix tree which supports indexing with part-of-speech (POS) tags. This allows our search algorithm to efficiently extract discontinuous multi-word targets without having to exhaustively enumerate all possible auxiliary words. Candidate generation methods typically produce many false positives. We address this problem by using a transformer model based on RoBERTa to filter out false-positive candidates, a deviation from past works which only used linguistic features for such filtering. Combined, our target identification model reaches an F1 of 0.775, an increase of +0.012 over the previous best-performing target identification model, achieving state-of-the-art performance.

Our frame identification model jointly encodes frame-semantic information with textual inputs in a similar manner to the previous best-performing multiple-choice classification model, FIDO (Jiang and Riloff, 2021). An important factor with multiple-choice classification is the necessity for negative samples in the set of choices due to its use of cross-entropy loss over each sample in the set, as opposed to the typical loss computation over each class. Because many targets only evoke a single frame, many of these sets of choices end up with only a single choice. In this case, roughly half of the training samples in FrameNet are unused or under-utilized due to their lack of negative samples. To address this, we randomly sample additional negative frames for each target that has less than N candidate frames. (N is empirically set as 4 in our experiments.) This change leads to an improvement in accuracy of +0.012 over FIDO's equivalent frame-only model, and +0.002 over FIDO's full model which utilizes both frame and lexical unit information while we only use frame information. This change also significantly improves the performance of our model on targets with only one candidate frame by +0.139, leading to an improvement in its ability to distinguish difficult under-utilized lexical units by +0.065 (Table 3). Furthermore, this improved the performance of rare frames with an improvement of +0.044 accuracy on frames with 5 or fewer samples compared to FIDO (Table 4).

In summary, our work makes the following contributions.

- We proposed a novel candidate target identification algorithm which extends coverage of candidate target generation methods to support discontinuous lexical units. Furthermore, we incorporated a RoBERTa model to filter out generated false-positive candidate targets.
- We addressed an important challenge in the standard multiple-choice frame identification model training methodology by sampling additional negative frames, enabling the model to learn from roughly half of the dataset which the model could not fully utilize.
- We derived two additional datasets to score the robustness of frame identification models— Test-1CF for evaluating the confidence of model predictions on lexical units which only evoke a single frame, and Test-UU for evaluating a model's ability to distinguish similar lexical units.
- We provided detailed and comprehensive experiments on target identification and frame identification, demonstrating the effectiveness of our approach including a +0.012 F1 improvement over the previous state-of-the-art model on target identification, a +0.014 accuracy improvement over FIDO's frame-only model on frame identification, and a +0.044 accuracy improvement on frames with 5 or fewer training samples compared to FIDO's full model.

The codebase of this work is made publicly available at `https://github.com/idirlab/frame`, including the source codes for the models and the scripts for preparing datasets.

2 Background and Related Work

FrameNet (Baker et al., 1998) is a collection of over 1,200 semantic frames, their respective inter-frame relationships, thousands of annotated sentences, and tens of thousands of exemplar sentences. Each frame describes a distinct semantic idea and can be evoked by different words or phrases called targets. A target is an instance of a lexical unit (LU), a distinct pairing of a word with a meaning which is represented by its lemma and POS, e.g., *think.v*. Frames are often evoked by several different LUs. For example, *discover.v*, *find.v*, and *invent.v* are LUs which can evoke the ACHIEVING_FIRST frame. Frame elements describe roles within a frame and are defined separately for each

frame, although commonly used frame elements with shared meaning may be found in multiple frames, e.g., Time and Agent.

Target identification is typically done by identifying potential LUs using the LU index provided in FrameNet, which maps each distinct LU to the frame it evokes, followed by further filtering them out using linguistic features (Das et al., 2014; Johansson and Nugues, 2007) or neural models (Swayamdipta et al., 2017). SEMAFOR (Das et al., 2014) is the most prominent rule-based generate-then-filter approach for target identification. SEMAFOR first generates a master list of all morphological variants of LUs within FrameNet, and then performs substring search on inputs using the master list to identify candidate targets. Then, SEMAFOR filters out targets using manually defined rules in addition to rules proposed by Johansson and Nugues (2007). More recent approaches (Bastianelli et al., 2020; Lin et al., 2021) have instead proposed span-based predicate classification methods which overcome the primary limitation of previous generate-then-filter methods, i.e., not being able to identify discontinuous LUs. While these approaches have strong performance, they rely heavily on the domain and quantity of their training data for the model to learn to distinguish predicate spans from non-predicate spans; however, FrameNet is known to have limited domain scope in the training data (Hartmann et al., 2017). Fortunately, candidate generation approaches avoid this problem due to their reliance on FrameNet's defined lexical units.

Frame identification is a much more studied task, especially in recent years. Approaches for frame identification can be generally categorized into graph-based classification methods (Lin et al., 2021; Bastianelli et al., 2020; Zheng et al., 2022a; Tamburini, 2022; Yan et al., 2023; Su et al., 2021) which utilize inter-frame relationships, syntactic methods (Swayamdipta et al., 2017; Das et al., 2014) which extract syntactic features to enhance classification models, and semantic methods (Jiang and Riloff, 2021; Zheng et al., 2022b) where semantic information about particular frames is encoded using pre-trained language models.

In our work, we explored semantic methods for two primary reasons. First, semantic methods benefit from pre-trained language models which have been exposed to and trained on billions of tokens and very diverse corpora. This enables more robust and generalized representations which support a wide variety of domains, an important feature when working with FrameNet. Second, semantic methods are easily extendable and scalable via replacing the language model with a larger model, if a particular application needs more performance, or a smaller model, if the application needs more throughput. On the contrary, graph-based methods are not necessarily as easily scalable due to their frequent reliance on computing intermediate representations for inter-frame relationships, which requires a significant data quantity.

Prior to our work, for frame identification the graph-based KGFI (Su et al., 2021) was the state-of-the-art model, and FIDO (Jiang and Riloff, 2021) was the best-performing semantic-based model, having performance on-par with KGFI. FIDO represents each target in a sentence as a multiple-choice classification task, wherein the model is optimized according to the highest-scoring target-frame pair. This is done by encoding textual representations of frame name, frame definition, LU name, and LU definition, along with the input sentence, into a pre-trained language model such as BERT (Devlin et al., 2019). Additionally, Zheng et al. (2022b) extended the use of semantic methods to argument identification, reaching state-of-the-art performance, further showing the value of semantic methods, particularly on zero- and few-shot tasks.

3 Target Identification

Target identification is the task of determining which tokens in a given sentence evoke a frame. For example, *thought* and *discovered* in Figure 1 evoke the frames OPINION and ACHIEVING_FIRST, respectively. Our target identification model, similar to SEMAFOR, consists of two steps: generating candidate targets, followed by filtering out false-positive targets.

3.1 Candidate Generation

An important factor which may be overlooked is the need for very high coverage in the candidate generation process as any missed true candidates will lead to error propagation on target identification, frame identification, and argument identification. SEMAFOR approached candidate generation by first building a master list of all morphological variants of LUs within FrameNet. For a given sentence, each n-gram of words in the sentence is then used to search the master list to identify candidate

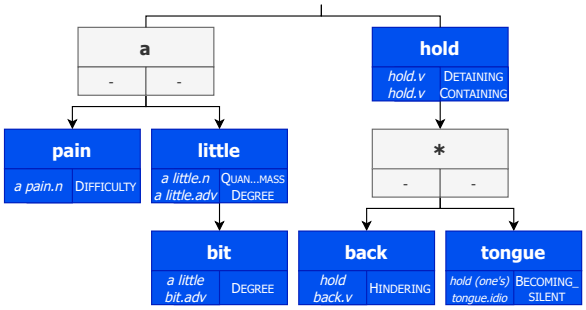

Figure 2: An example of the lexical unit tree generated from some of the lexical units defined in FrameNet. Each colored (blue) node indicates a group of lexical units that share the same tokens, as traversed from the root of the tree. The left box within a node represents the particular lexical units, with their corresponding frames on the right.

targets. This approach is unable to handle discontinuous LUs as it would require the master list to contain all possible variants of each discontinuous LU, a task which becomes infeasible as we support more auxiliary words. For example, *hold back.v* could appear in a sentence as "held back", "hold them back", or "hold Bob back".

We approach candidate generation from the opposite direction, starting from each token in an input sentence and attempting to search for an LU which it could belong to. To handle discontinuous LUs such as the example above, we allow certain multi-word LUs to contain wildcard tokens between them, e.g., pronouns and proper nouns separating a verb and its article in a phrasal verb. To enable efficient searching for lexical units using words in a sentence, we created a lexical unit tree by using a prefix tree similar to the structure in Aho and Corasick (1975). The Aho-Corasick algorithm involves building a finite-state machine that resembles a prefix tree and allows transitions between any two nodes, not just from parent to child nodes. In the Aho-Corasick algorithm, nodes are determined by each character in a string; however, for our lexical unit tree, each node represents a word or POS wildcard instead.

3.1.1 Lexical Unit Tree

We construct the lexical unit tree using FrameNet's lexical unit-frame mapping such that each token within a lexical unit belongs to its own node. An example of this can be found in Figure 2. Much like a standard prefix tree, subsequent tokens become child nodes to previous tokens. To maintain the LU-frame mapping, each node contains the set of frames evoked by the lexical unit lemma produced by the sequence of tokens from the root. Hence, in Figure 2, *a* → *pain* can evoke the frame DIFFICULTY (*a pain.n*), while *a* → *little* can evoke the DEGREE frame (*a little.adv*) and the QUANTIFIED_MASS frame (*a little.n*).

To support discontinuous LUs, we incorporate certain POS tags, namely PRON, PROPN, CCONJ, DET, and NOUN, as children for any multi-word verb LU in the form of a wildcard token (∗). This enables searching the tree with POS tags to handle phrasal verbs which occur frequently in English. For example, in Figure 2, *hold back.v* will be represented as *hold* → ∗ → *back*, where ∗ can be any word with one of the aforementioned POS tags. Using this method, we are able to identify simple discontinuous targets, such as "write something down" (*write down.v*) and "wrap it up" (*wrap up.v*), or even more complex targets, such as "turn your friends in" (*turn in.v*), neither of which SEMAFOR would be able to identify.

Searching the lexical unit tree is similar to searching a standard prefix tree with one significant exception: child nodes can also be indexed using POS tags. Generating candidate targets is an iterative process. Given a sentence, our search method traverses through each token in the sentence, attempting to match them to a node in the tree starting from the root. If a node is found, the node is maintained in a list of potential LUs. Then, the next word will attempt a match with all potential LU children in addition to starting from the root. Any time a match is found, if the node has potential frames, we mark the token span as a candidate target for them. To attempt the match, we use each of the following features of a given word: the word as it exists in the sentence, the normalized token, the lemma (with and without POS context), the lexicalized form of numbers, the expanded form of contractions, the unhyphenated form of hyphenated words, and the POS tag of the word. We include the pseudocodes of the LU tree search (Algorithm 1) and candidate generation (Algorithm 2) algorithms in Appendix A.

3.1.2 Complexity

One important factor of using a prefix tree architecture is its high efficiency, especially when compared to exhaustively searching a given string for all morphological variants for each lexical unit, as is done by Das et al. (2014). Building our lexical unit tree is a one-time process with a time complex-

ity of $\mathcal{O}(L * W)$, where L is the total number of LUs and W is the average number of words per LU. In FrameNet 1.7, W is roughly 1.1 and L is 13,572. To search the lexical unit tree for a single LU using the steps mentioned above, the time complexity is $\mathcal{O}(m * W)$, where $m = 7$ is the constant number of features used when searching the lexical unit tree, as discussed in the end of Section 3.1.1. Searching an entire sentence as discussed above has a time complexity of $\mathcal{O}(n * m * W)$, where n is the number of words in the sentence.

3.2 Target Filtering Model

To filter out false-positive candidate targets, we employ a RoBERTa-based binary classification model. This approach removes the need to filter out candidates using manually defined rules which are prone to noise, do not evolve with language, and are practically infeasible for exhaustiveness. Consider a sequence of contextualized embeddings $\boldsymbol{x}$ and its target span embedding $\boldsymbol{x}_{i:j} = \sum_{n=i}^{j} x_n$, where i and j are the start and end positions of the target span, respectively. We optimize the model by minimizing the binary cross-entropy loss:

$$\mathcal{L} = -y \cdot \log(\sigma(z_{i:j})) - (1-y) \cdot \log(1-\sigma(z_{i:j})) \quad (1)$$

where $z_{i:j}$ is the model's predicted class score for the target span $\boldsymbol{x}_{i:j}$ and y is the ground-truth label indicating if the candidate span is a real target.

In addition to the model above, we also experimented with unifying our models by using our frame identification model as a proxy for this filtering step, i.e., instead of generating candidate targets, filtering the targets, and then identifying the frames, we would use the frame identification model (Section 4) to classify the candidate targets directly. We experiment with this approach in Section 5.3.

4 Frame Identification

Frame identification is the task of identifying which frame is evoked by a given target. A target is a single word in most cases but, as discussed in Section 3, can also be a span of multiple consecutive or nonconsecutive words. Furthermore, a sentence may have multiple frame-evoking targets or even multiple occurrences of a frame.

4.1 Lexicon Filtering

There are over 1,200 frames in FrameNet. Given a sentence and an identified target in the sentence, in most cases, we can exclude the majority of frames by recognizing that a target, which is an instance of a particular LU, can only evoke certain frames. This idea, coined *lexicon filtering* (Su et al., 2021), has been used in most previous works on target identification and frame identification. We found that each lexical unit lemma defined in FrameNet's lexical unit index can evoke on average 1.29 possible frames. In the fulltext annotations, the number is higher at 2.12 candidate frames per annotation instance of lexical unit lemma due to a heavy class imbalance in the annotations.

4.2 Frame and Lexical Unit Definitions

Recent work has shown that incorporating the definitions of frames and LUs provided in the FrameNet dataset into frame identification models results in improved performance (Jiang and Riloff, 2021; Su et al., 2021). This is particularly useful for frames with very few samples in the fulltext annotations, which are the overwhelming majority of frames, as it allows the model to utilize additional information in its classification. As an example of an LU's definition, the OPINION frame's LU *think.v* is defined as "to have a particular opinion".

4.3 Task Formulation

We follow the approach used by FIDO in representing the frame identification task as a multiple-choice classification task among candidate frames produced from lexicon filtering. However, we identified an important problem in FIDO's implementation, specifically for targets with only a single candidate frame. Let F^c be the set of candidate frames evoked by target $\boldsymbol{x}_{i:j}$. FIDO computes cross-entropy loss using softmax as follows:

$$P(f|\boldsymbol{x}_{i:j}) = \frac{\exp(g(\boldsymbol{x}_{i:j}, f))}{\sum_{k=1}^{|F^c|} \exp(g(\boldsymbol{x}_{i:j}, F^c_k))} \quad (2)$$

where $g(\boldsymbol{x}, f)$ is a learned classification function which takes a sequence of tokens $\boldsymbol{x}$ and a frame f as inputs. The problem arises when $|F^c| = 1$, i.e., $F^c = \{f^*\}$ and thus $F^c_1 = f^*$ where f^* is the true frame. For such a target $\boldsymbol{x}_{i:j}$, Equation 2 results in a value of 1 regardless of the output of $g(\boldsymbol{x}_{i:j}, f)$, because

$$P(f^*|\boldsymbol{x}_{i:j}) = \frac{\exp(g(\boldsymbol{x}_{i:j}, f^*))}{\exp(g(\boldsymbol{x}_{i:j}, F^c_1))} = 1,$$

which leads to a computed cross-entropy loss of 0 because $-\log(1) = 0$.

To address this problem, during the training phase for each target we randomly sample a number of additional frames (i.e., negative examples) so that the target gets N samples in total. We hypothesize that this will benefit the model in two ways. First, by adding negative samples, the loss function is able to utilize the positive sample. Second, a negative sample—a pair formed by a given target and a frame—provides not only one extra false candidate frame for the target but also an additional negative example for the frame, potentially improving the model's performance on rare frames.

Note that, as a result of including these negative samples, we only encode frame definitions into the input as we speculate including lexical unit definitions would trivialize the task, i.e., the model may learn to match the target token (e.g., "thought") with the positive LU's lemma (e.g., *think.v*). Future work is needed to integrate lexical unit definitions into the input.

A simpler alternative option would be to treat frame identification as a binary classification task. However, such an approach would not have the aforementioned benefits. To test this hypothesis, we also implemented our model using binary cross-entropy loss (further discussed in Section 5.3).

4.4 Model Definition

Like FIDO, we represent frame identification as a multiple-choice classification task, wherein the model is optimized to predict which of a particular set of frames is the best choice for a given target. Formally, we present the frame identification task as follows. Given a sequence of contextualized embeddings $\boldsymbol{x}$, a target $\boldsymbol{x}_{i:j}$, a set of candidate frames F^c produced via lexicon filtering, and a disjoint set of $N - |F^c|$ randomly sampled frames F^r, identify the frame $f \in F' = F^c \cup F^r$ evoked by the target $\boldsymbol{x}_{i:j}$. Like FIDO, we create a separate input for each target-frame pair in a given sentence.

To produce the contextualized embeddings $\boldsymbol{x}$ for a given frame f, we use $\text{BERT}_{\text{base}}$ to encode the input sequence s_f, created by concatenating the frame name and definition to the end of the input sentence, separated by BERT's [SEP] token. Given that there can be multiple frames in a sentence, our model scores the representation using target token $\boldsymbol{x}_{i:j}$, as it allows the model to focus on a single target at a time rather than the entire sentence at once. Finally, we optimize the model by minimizing the cross-entropy loss:

$$\mathcal{L} = -\sum_{k=1}^{|F'|} y_k \cdot \log(P(F'_k|\boldsymbol{x}_{i:j})) \quad (3)$$

where y_k is the ground-truth label indicating if the frame F'_k is indeed evoked by $\boldsymbol{x}_{i:j}$.

5 Experiments

For evaluating the effectiveness of our system, we conducted experiments to answer the following questions: **Q1)** *What is the coverage of our candidate generation model for target identification?* As discussed in Section 3.1, maintaining a very high coverage is crucial due to error propagation on downstream tasks. **Q2)** *Are contextualized embeddings beneficial for filtering candidate targets in target identification models?* Previous approaches have found success with manual rules for candidate filtering; however, no previous works have studied the impact of pretrained language models on candidate target filtering. **Q3)** *Do additional negative frames improve frame identification model performance on under-utilized samples?* As discussed in Section 4, we hypothesize that FIDO does not effectively use all of the samples in the dataset due to the lack of negative training samples on many of the frames. To verify our hypothesis, we must evaluate the impact of additional negative training samples on the under-utilized samples. **Q4)** *Does additional negative sampling improve frame identification performance on rare frames?* Similar to Q3, because our negative sampling strategy pulls uniformly from all frames while FrameNet is not uniformly balanced, we must verify whether this strategy is beneficial for rare frames. **Q5)** *Can our frame identification model substitute the target filtering model?* Because target filtering and frame identification models perform similar functions, we aim to evaluate whether we can use our frame identification model to directly filter candidate targets.

5.1 Datasets

We evaluate our models using FrameNet 1.7 (Ruppenhofer et al., 2016) with the standard train, validation, and test split defined by Swayamdipta et al. (2017), resulting in 19,391 training samples, 2,272 validation samples, and 6,714 test samples. To compare our system with previous approaches, we also train and evaluate on FrameNet 1.5 which has 15,017 training samples, 4,463 validation samples, and 4,457 test samples.

5.2 Target Identification

To evaluate the coverage of our candidate generation model (**Q1**), we compute the percentage of targets from the test set that our candidate generation model identifies, i.e., the recall. This number is important as it gives us an absolute upper bound recall for our target identification model, thus we would like to maximize it. Additionally, we evaluate the precision of our candidate generation model, as we need to ensure that it does not simply mark every token as a candidate. The precision measure gives us an absolute lower bound precision for our target identification model. In theory, one could select all words in each sentence as candidate targets, resulting in 100% recall; however, this would be no different from simply applying a statistical classifier to each token, which Das et al. (2014) found to be ineffective.[1] Therefore, it is important that we find a balance which maintains a very high recall without sacrificing precision when generating candidate frames.

To evaluate the impact of using contextualized embeddings in our target filtering model (**Q2**), we compare our filtering approach with SEMAFOR and our own automatically identified manual filters.[2] Furthermore, we also compare our target identification model with two modern approaches, Bastianelli et al. (2020), an end-to-end system which encodes constituency information through a graph convolutional network (Kipf and Welling, 2016) and models target identification as a sequence labelling task by employing a conditional random field layer (Lafferty et al., 2001), and Lin et al. (2021), another end-to-end graph-based system which enumerates possible spans and utilizes BERT to filter out non-predicate spans. Additionally, we include Swayamdipta et al. (2017) which employs a basic target identification model that learns a classifier using a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to classify lemmas belonging to lexical units in FrameNet. The results of these experiments can be found in Table 1.

[1]We verified this finding using a binary BERT classifier.

[2]We used a conservative filter to remove candidate targets using linguistic features of each target, including part-of-speech tags, dependency tags, and lemmatized and normalized tokens. These linguistic features were obtained using spaCy (Honnibal et al., 2020).

Model	FN1.5	FN1.7
Das et al. (2014)	0.454	-
Swayamdipta et al. (2017)	0.732	0.733
Bastianelli et al. (2020)	0.768	-
Lin et al. (2021)	0.769	0.763
Our model	**0.773**	**0.775**
Our model (manually filtered)	0.388	0.392

Table 1: F1 score of our target identification model compared against previous approaches.

Model	FN1.5		FN1.7	
	All	Amb	All	Amb
Das et al. (2014)	0.836	0.692	-	-
Hermann et al. (2014)	0.887	0.737	-	-
Hartmann et al. (2017)	0.876	0.738	-	-
Yang and Mitchell (2017)	0.882	0.757	-	-
Swayamdipta et al. (2017)	0.864	-	0.866	-
Peng et al. (2018)	0.900	0.780	0.891	0.775
Bastianelli et al. (2020)	0.901	-	-	-
Lin et al. (2021)	0.906	-	0.906	-
Su et al. (2021)*	0.919	0.823	**0.924**	**0.844**
Tamburini (2022)*	**0.922**	**0.831**	0.922	0.843
Zheng et al. (2022a)	0.917	-	-	-
Jiang and Riloff (2021)	0.913	0.810	0.921	0.836
Jiang and Riloff (2021) (frame)	0.901	-	0.911	-
Our model (binary)	0.877	0.785	0.887	0.816
Our model	0.917	0.818	0.923	0.841

* Performance can not be verified due to private source code.

Table 2: Performance of our model compared with several other frame identification models. Ambiguous (Amb) refers to the subset of targets within the test set which have more than one possible frame. The best performing model is bolded and the better model between ours and FIDO (Jiang and Riloff, 2021) is underlined.

5.3 Frame Identification

We experimented with two approaches to solve the under-utilized sample problem discussed in Section 4. First, we experimented with randomly sampling additional frames for each target during the training phase. As Jiang and Riloff (2021) found, multiple-choice classification is a promising approach for frame identification; however, it needs negative samples to function and not all targets have negative samples. Additionally, we attempted to represent the task as a binary classification task instead, maintaining the same inputs, and computing the binary cross-entropy loss over each sample instead of one-hot encoded cross-entropy loss over each target. The binary classification task is presented as *Our model (binary)* alongside our first approach in Table 2.

We evaluate the benefits of using negative samples by comparing our frame identification model's predictions with FIDO's predictions on the test

Model	Test-1CF	Test-UU
FIDO (Jiang and Riloff, 2021)	0.754	0.538
Our model	**0.893**	**0.603**

Table 3: Accuracy of our frame identification model on Test-1CF, a subset of the test dataset containing targets that only have one candidate frame, and Test-UU, an augmentation of Test-1CF with additional difficult negative samples for the evaluation of multiple-choice classification models.

K	# Frames	Our Model	FIDO	Δ
1	94	**0.781**	0.753	+0.028
3	235	**0.810**	0.778	+0.032
5	316	**0.853**	0.809	+0.044
10	426	**0.850**	0.826	+0.024

Table 4: Accuracy of our frame identification model compared with FIDO on frames which only appear K or fewer times in the FrameNet 1.7 training set.

samples which only have one possible frame (**Q3**). The goal of this is to see whether adding negative samples improves the model's ability to identify these samples, or if it is essentially acting as a rule-based filter, i.e., when there is only one option, picking it regardless of the score. To do this, we create a subset of the test dataset, Test-1CF, containing all test samples with only one candidate frame (2,672 total), i.e., $|F^c| = 1$, and we evaluate each model with a softmax classification threshold of 0.5 (Table 3).

To evaluate the performance of our model on under-utilized samples, we further build on Test-1CF by sampling 3 additional negative LUs from the same frame which have the same POS tag. If we cannot retrieve 3 samples, we randomly sample the remaining LUs from all LUs with the same POS tag. This new dataset, Test-UU, forces the model to differentiate between similar LUs, a challenge for any model that does not handle under-utilized samples correctly. Test-UU contains 2,672 positive samples and 8,016 negative samples, totaling 10,688 samples. Finally, the model is evaluated as a multiple-choice classification task (Table 3).

To evaluate the effects of additional negative sampling on rare samples (**Q4**), we compare the accuracy of our frame identification model with FIDO on frames which only appear 1, 3, 5, and 10 or fewer times within the FrameNet 1.7 training set. The resulting metrics can be found in Table 4.

Finally, we experiment with replacing the intermediate target filtering step by directly classifying candidate targets (**Q5**). To do this, we run the frame identification model directly on the candidate targets produced from Section 3.1 and, if the frame identification model identifies a frame, we mark the candidate target as a true target. If our model does not predict any frames for a particular target with positive logits, we remove the candidate target. This also introduces a new task for future works to compare with as we believe the future goal of frame-semantic parsing is to remove the 3-step approach (target identification, frame identification, and argument identification) and unify all sub-tasks into a single task. We compare our model with FIDO in Table 5.

Model	Acc	F1
Our model (candidate filter)	**0.788**	**0.775**
FIDO (Jiang and Riloff, 2021)	0.653	0.644
Our model	0.664	0.678

Table 5: Performance of frame identification models on the candidate target filtering task using our generated candidate targets.

6 Results and Discussion

6.1 Target Identification

Q1. We found that our candidate generation model is able to identify targets with a recall of 0.994 and a precision of 0.145. This indicates that, on the one hand our LU tree is able to cover 99.4% of the targets in the test dataset, but on the other hand most of the generated candidates are false-positives. The candidate generation model strikes a sufficiently good balance between maximizing coverage and minimizing excess candidates. We identified several instances where the model produces candidate targets for frames which do not appear in the fulltext annotations. While this negatively impacts the performance on the FrameNet dataset, we choose not to remove these via postprocessing as it could be detrimental to the model's out-of-domain capabilities.

Q2. Using manual filters, as described in Section 3, we achieved an F1 score of 0.392 (Table 1). These results form a baseline for our candidate generation algorithm to compare the efficacy of utilizing RoBERTa to filter out candidate targets. It is possible that using more robust rules can lead to a higher F1 score, such as the rules used by Das et al. (2014). However, this was not the focus of our work. In this case, SEMAFOR acts as an even stronger baseline for manual candi-

date filtering. Filtering the candidate targets using our RoBERTa-based filtering model showed much more promise, achieving an F1 score of 0.775 on target identification, an improvement of +0.012 over Lin et al. (2021), the previous-best target identification model.

6.2 Frame Identification

Overall, we found that our frame identification model performed similar to more recent models, achieving an accuracy of 0.923 on FN1.7 and 0.841 on ambiguous samples (Table 2). Notably, our model which only uses frame information performs slightly better than FIDO which uses frame and LU information. Additionally, our model achieves an accuracy of +0.012 higher than FIDO's frame-only model. We also found that tackling the task as binary classification did not perform as well as using additional negative samples. These results show that using negative samples can improve the model's overall performance.

Q3. Under-utilization of samples is the most important problem we aimed to solve in this work. We found that, on Test-1CF, FIDO predicted the correct class with a positive classification score only 75.4% of the time (Table 3). Thus, the other 24.6% of the samples were predicted incorrectly but were classified correctly because of the lexicon filtering step. Meanwhile, our model predicted positive classification scores for 89.3% of the samples, indicating our negative sampling enabled the model to identify more samples without relying on lexicon filtering. We also found that on Test-UU, where differentiating the negative samples is much harder as they belong to the same frame and/or POS, our model achieved a +0.065 improvement of accuracy.

Q4. We found that using additional negative frames resulted in an improvement on rare frames at each tested threshold. As we expected, our model performs significantly better on frames with limited training samples. In particular, frames with 5 or fewer samples in the training set saw an improvement of +0.044, as shown in Table 4. This finding shows that additional negative sampling is a necessity for multiple-choice classification tasks as it significantly improves the performance on frames that have few samples in the training set. We also see that, at 10 samples, FIDO begins to close the performance gap, reducing the improvement to just +0.024. This further supports our claim that additional negative samples improve the performance on frames with few samples.

Q5. Applying our frame identification model on our candidate targets resulted in lower accuracy and F1 score when compared to our target identification model (Table 5). We hypothesize that this is likely due to the difference in training tasks as the frame identification model is trained under the assumption that the given target is in fact a real target and may bias its predictions accordingly, while the candidate target filtering model is tasked with determining whether a given target is in fact a real target. Interestingly, our model still outperformed FIDO on the candidate filtering task, indicating that negative sampling has value outside of just frame identification. While the performance is not yet competitive, it shows promise that future works on multiple-choice classification will be able to merge the candidate target filtering and frame identification tasks, much like Zheng et al. (2022b) did for frame and argument identification.

7 Conclusion

In this work, we presented a novel approach for candidate target generation which utilizes a modified prefix tree to extend support to discontinuous lexical units, resulting in a 99.4% coverage on FrameNet annotations. Following this, our model filters out false-positive candidates, resulting in performance exceeding the previous state-of-the-art solution on target identification. Additionally, we identified and solved an important issue with a previous multiple-choice classification approach, resulting in increased performance, particularly on frames with few samples and targets which only evoke a single frame. Finally, we explored incorporating the frame identification model into the candidate target filtering task, and found promising results, indicating that future works may be able to merge candidate target filtering, frame identification, and argument identification into one step.

Limitations

Lexicon filtering is very helpful for in-domain lexical units; however, it may lead to problems with regional terms and slangs. While we did not experiment with any out-of-domain lexical units, it is important for future works to recognize this as a limitation in frame identification. Additionally, as mentioned in Section 4.3, we only encode frame definitions into the inputs as including LU informa-

tion would trivialize the task. Future work will be needed to address this challenge.

Our work was done based on the original FrameNet and under the assumption that the inputs are in English. While developing our model, we did not consider other versions of FrameNet, such as the Chinese, French, or Japanese versions, and we cannot ensure the linguistic tools we used will be applicable to other languages.

Ethics Statement

We have not taken the opportunity to assess the existence of biases in the FrameNet dataset which could inadvertently affect our system. Applications of our system could lead to potential biases for or against certain genders, races, or ethnicities due to underlying biases in the training data. In particular, it is known that FrameNet uses annotations from materials which mention chemical and nuclear weapons, and regions such as North Korea, Iran, Syria, and Taiwan.

We believe further improvement on frame-semantics can produce more explainable results for down-stream applications and assist in projects which have positive societal impacts.

Acknowledgements

This work is supported by the National Science Foundation under Grants 1719054, 1937143, and 2333834. We also extend our gratitude to the Texas Advanced Computing Center (TACC) for providing compute resources used in this work's experimentation.

References

Alfred V. Aho and Margaret J. Corasick. 1975. Efficient string matching: an aid to bibliographic search. *Commun. ACM*, 18(6):333–340.

Fatma Arslan, Josue Caraballo, Damian Jimenez, and Chengkai Li. 2020. Modeling factual claims with semantic frames. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 2511–2520, Marseille, France. European Language Resources Association.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada. Association for Computational Linguistics.

Emanuele Bastianelli, Andrea Vanzo, and Oliver Lemon. 2020. Encoding syntactic constituency paths for frame-semantic parsing with graph convolutional networks. *CoRR*, abs/2011.13210.

Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Charles J. Fillmore and Collin Baker. 2009. 313 A Frames Approach to Semantic Analysis. In *The Oxford Handbook of Linguistic Analysis*. Oxford University Press.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Silvana Hartmann, Ilia Kuznetsov, Teresa Martin, and Iryna Gurevych. 2017. Out-of-domain FrameNet semantic role labeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 471–482, Valencia, Spain. Association for Computational Linguistics.

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. 2014. Semantic frame identification with distributed word representations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1448–1458, Baltimore, Maryland. Association for Computational Linguistics.

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

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength natural language processing in python.

Tianyu Jiang and Ellen Riloff. 2021. Exploiting definitions for frame identification. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2429–2434, Online. Association for Computational Linguistics.

Richard Johansson and Pierre Nugues. 2007. LTH: Semantic structure extraction using nonprojective dependency trees. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 227–230, Prague, Czech Republic. Association for Computational Linguistics.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*.

ZhiChao Lin, Yueheng Sun, and Meishan Zhang. 2021. A graph-based neural model for end-to-end frame semantic parsing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3864–3874, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Hao Peng, Sam Thomson, Swabha Swayamdipta, and Noah A. Smith. 2018. Learning joint semantic parsers from disjoint data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1492–1502, New Orleans, Louisiana. Association for Computational Linguistics.

Josef Ruppenhofer, Michael Ellsworth, Miriam R. L Petruck, Christopher R. Johnson, Collin F. Baker, and Jan Scheffczyk. 2016. FrameNet II: Extended theory and practice. *International Computer Science Institute, Berkeley, California.*

Yuqi Si and Kirk Roberts. 2018. A frame-based nlp system for cancer-related information extraction. *AMIA annual symposium proceedings*, 2018:1524–1533.

Anders Søgaard, Barbara Plank, and Héctor Martínez Alonso. 2015. Using frame semantics for knowledge extraction from Twitter. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Evangelia Spiliopoulou, Eduard Hovy, and Teruko Mitamura. 2017. Event detection using frame-semantic parser. In *Proceedings of the Events and Stories in the News Workshop*, pages 15–20, Vancouver, Canada. Association for Computational Linguistics.

Xuefeng Su, Ru Li, Xiaoli Li, Jeff Z. Pan, Hu Zhang, Qinghua Chai, and Xiaoqi Han. 2021. A knowledge-guided framework for frame identification. 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)*, pages 5230–5240, Online. Association for Computational Linguistics.

Swabha Swayamdipta, Sam Thomson, Chris Dyer, and Noah A. Smith. 2017. Frame-semantic parsing with softmax-margin segmental rnns and a syntactic scaffold. *arXiv*, abs/1706.09528.

Fabio Tamburini. 2022. Combining ELECTRA and adaptive graph encoding for frame identification. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 1671–1679, Marseille, France. European Language Resources Association.

Zhichao Yan, Xuefeng Su, Qinghua Chai, Xiaoqi Han, Yunxiao Zhao, and Ru Li. 2023. Multiple pos dependency-aware mixture of experts for frame identification. *IEEE Access*, 11:25604–25615.

Bishan Yang and Tom Mitchell. 2017. A joint sequential and relational model for frame-semantic parsing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1247–1256, Copenhagen, Denmark. Association for Computational Linguistics.

Ce Zheng, Xudong Chen, Runxin Xu, and Baobao Chang. 2022a. A double-graph based framework for frame semantic parsing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4998–5011, Seattle, United States. Association for Computational Linguistics.

Ce Zheng, Yiming Wang, and Baobao Chang. 2022b. Query your model with definitions in FrameNet: An effective method for frame semantic role labeling. *arXiv*, abs/2212.02036.

A Pseudocode

Algorithm 1 Lexical Unit Tree Search

```
1:  procedure SEARCHLUTREE(root, query)
2:    current_node ← root
3:    for token in query do
4:      if token ∈ current_node.children or token.pos ∈
      current_node.children then
5:        current_node ← current_node.children[char]
6:      else
7:        return null
8:    if current_node.is_end_of_word then
9:      return current_node.frames
10:   else
11:     return null
```

Algorithm 1 is the modified LU tree search algorithm containing the standard method with the additional condition on Line 4 to search for POS tags in the child nodes.

Algorithm 2 is the linear-time (with respect to the length of the input) method to generate candidate targets using our LU tree based on a given tokenized input sentence. Note that this algorithm

Algorithm 2 Candidate Target Generation

```
procedure GenCandidates(sentence, lu_tree)
  candidates ← {}
  potentials ← {}
  for token in sentence do
    for p in potentials do
      if token ∈ p.children then
        potentials.Insert(p.children[token])
      else if token.pos ∈ p.children then
        potentials.Insert(p.children[token.pos])
      if p.frames ≠ null then
        candidates.Insert(p)
      potentials.Remove(p)
    if token ∈ lu_tree.children then
      candidate.Insert(token)
      if lu_tree[token].has_children then
        potentials.Insert(lu_tree[token])
  return candidates
```

will be run an additional time for each token feature used, e.g., token lemma, normalized form, etc., as mentioned in Section 3.1.1.

B Reproducibility

Our code is publicly available for use at https://github.com/idirlab/frame. Due to FrameNet's data distribution policy, we are unable to re-publish the Test-1CF and Test-UU datasets. However, our repository provides the necessary scripts for reproducing the datasets using FrameNet.

B.1 Hyperparameters

Our target and frame identification models are built using PyTorch with the Hugging Face library.[3] We use Hugging Face's implementation of the pretrained BERT and RoBERTa models. We trained our models using several GPUs, including GTX 1070, GTX 1080 Ti, GTX 2080 Ti, and NVIDIA Tesla T4. Each GPU is able to train either of our models within a few hours, although some batch size configurations may be necessary to fit everything into memory. Our final models were trained a single time and we did not explore any hyperparameter options.

B.1.1 Target Identification Model

We optimize the weights of our target identification model for 3 epochs using AdamW (Loshchilov and Hutter, 2017) with a learning rate of $5e-5$ (cosine LR scheduler) and a batch size of 36.

B.1.2 Frame Identification Model

We optimize the weights of our frame identification model for 5 epochs using AdamW with a learning rate of $2e-5$ and a batch size of 16. To ensure we have sufficient space for the input sequence, frame definition, and lexical unit definition, we use a maximum sequence length of 300 tokens.

[3]The Hugging Face library can be found at https://huggingface.co/.