Inherent Semantic Relation Labeling: A Deep Hierarchical Approach

Chao Chen, **Jiajing Chen** and **Da Li**
Computer Science Department
New York University
{cc7287, jc12020, dl4593}@nyu.edu

***Abstract*—Semantic labeling classification is important yet challenging in natural language processing(NLP) tasks. To tackle such a multiple-categories classification problem, BERT is one of the classic approaches by using a mask autoencoder-decoder structure. However, BERT cannot integrate other useful tagging-related properties, such as BIO and POS tags. Thus, we introduce a hybrid deep-learning-based system BERT++ to build a semantic relation labeling system to find multiple types of Partitive Nouns in NomBank[1]. BERT++ integrates BERT and human-understandable labels in the pipeline. Combining word embedding with rich semantic information and manual labels of POS(part of speech)[2] and BIO(Begin/Inside/Other Chunk tag)[3] for constituent-based and dependency-based representations respectively, we achieved slightly better classification performance compared to the Vanilla BERT. The code we used to train and evaluate our models is available at https://github.com/JiajingChan/NLP-final.**

***Keywords*—Semantic Relation Labeling, Deep Bidirectional Transformers**

I. Introduction

The Semantic Role Labeling (SRL) task focuses on 1) mining and using the correlations among labels, and 2) extracting the critical information of corresponding labels from original documents. Our intuition is to use the semantic information of labels as guidance to capture important fine-grained document information.

An innovative mechanism–adjustive attention is proposed in this paper to calculate the semantic relationship between word and label explicitly. It generates label-special word representation. Compared with normal attention methods, the focus area of the adjustive attention mechanism becomes more meaningful and discriminatory.

In the computational linguistics field, there has been a major transition from developing task-specific models built from scratch to fine-tuning approaches based on large general-purpose language models [4]. Currently, the most commonly used pre-trained model of this type is BERT[5]. This model and its derivatives are based on the transformer architecture. Many state-of-the-art results on benchmark natural language processing (NLP) tasks have been improved by fine-tuned versions of BERT and BERT-derived models.

BERT (Devlin et al., 2018)[5]is a large-size language representation model trained on a large corpus of English text that can achieve state-of-the-art results on various natural language processing tasks. BERT came a decade later than the release of NomBank, and there's been little previous work using it to conduct NomBank-based tasks and related experiments. However, as powerful as BERT is, its large parameter size makes it significantly more computationally expensive to train and use.

We are focusing on a deep-learning-based method to find ARG0-ARG2, Predicate, and Support of Partitive Nouns in NomBank. Statistical learning models with human-designed features were learned using labeled/mined data. The statistical learning method significantly improved many NLP tasks, particularly in Machine Translation and search engine technology[6]. Recently, Deep-learning shows great potential in natural language task[7], especially when this trend is sparked by the success of word embedding[8], [9] and deep learning methods[10]. Among all deep learning methods, since transformer[11] has achieved promising performance, BERT[5] has become one of the classic methods in Natural language processing. BERT is designed to pre-train deep bidirectional representations from the unlabeled text by conditioning across all layers on both bidirectional contexts at the same time. And BERT is both conceptually simple and empirically powerful in understanding the sentence's semantic meaning.

With our approach BERT++, which uses BERT as a baseline and incorporates additional elements in the pipeline, we should make a good balance between precision and recall with both POS and BIO. Besides, the token and sentence number are excluded from our selected feature because our research interest focuses on how those traditional linguistic labels help in nowadays deep hierarchical black boxes.

To sum up, we achieve the following contributions in this paper:

1. We proposed a hybrid BERT model leveraging the NomBank-based semantic role labels and the uncased pre-trained BERT with the merits of both the explainability from traditional statistical learning and human-designed features and the precision that Bert brings fused with other features.

2. This paper sheds light on ARG0-ARG2, Predicate, and Support provided in the NomBank because they occur most frequently with the greatest research attention; it focuses on partitive nouns (nouns that are used to describe a part or quantity of something), and those popular tags provide heuristical insights for the other semantic labels of interest.

3. In addition to utilizing BERT as a baseline, we make efforts to augment the plain BERT model with classic linguistic labels of POS(part of speech) and BIO(Begin/Inside/Other Chunk tag) for our specific focus on inherent semantic relationships identification and extraction in token granularity.

The remainder of this paper is organized as follows: Section II introduces related works about multi-label classification and label embedding methods. Section III describes the BERT++ model in detail. Section IV gives extensive experiments to validate the effectiveness of our approach. Discussion for our conclusion is given in section V.

II. Related Work

Semantic Relation Labeling (SRL)[12] is one of the cornerstones of computational linguistics research and natural language understanding systems. For computers to make effective use of information encoded in text, it is essential to detect the events that are being described and the event participants.

1) Lexical Resources
   Though it seems fairly straightforward, it took a long way for researchers to come up with a uniform structure. For instance, English allows several different syntactic constituents to present the same semantic role, and several different semantic roles to be presented by the same syntactic constituent.
   The advent of predicate-argument structure has enabled the creation of large and unified annotated corpora. It's vital and universally used to train automatic linguistic systems with structured automatically-annotated lexicons. The universally used Semantic Roles is shown in Table I [12].
   Despite the intuitive appeal of the case studies, the semantic role has failed to build a consensus around a set of examples or determining rules. The general agreement could merely be reached on straightforward cases. There are still substantial disagreements on exactly what and how to assign role labels to words.
   There are many universally recognized roles definitions for SRL tasks, like FrameNet[13], VerbNet[14] and PropBank[15]. Our solution to this disparity is to select and follow the definitions and examples derived from NomBank[1]. The NomBank project provides argument structures for instances of common nouns in the Penn TreeBank II corpus. As a part of the larger effort to annotate the Penn TreeBank II corpus, NomBank leads to better tools for automatically analyzing text with detailed specifications and creating processes.
2) Learning-based Methods
   Machine learning attempts are also made to solve this classic linguistic problem of SRL. In the Machine Learning context, SRL can be treated in the general framework of multi-classification tasks[16]. For a given word (token), together with its corresponding constituent in parse, the model is to pick the most likely label from a pre-defined set of semantic role labels. Thus, the commonly essential design for semantic role labeling systems is to extract features of each word/token given the background sentence it belongs to. The early work[17] used a backoff lattice, in which probabilities of a label given various subsets of the features were calculated directly from counts in the training data. However, the backoff lattice approach does not scale to a larger corpus with more features. Thus,[18] proposed a more general-purpose to use machine-learning algorithms: a decision tree to gain generalizability.
   In a side-by-side comparison, both[19] and [20] used a maximum entropy classifier, a.k.a., logistic regression to improve the performance compared to the backoff lattice solution. Besides, [21] used support vector machines(SVMs) with the same features to yield a 10% improvement. Overall, these attempts in semantic role labeling used discriminative approaches to exploit large scaled corpus, and gain improvements compared to direct frequency-based models such as lattice backoff and decision trees. The performance also varies from task to task. In some cases, the SVMs achieve better performance than maximum entropy, but the improvement is often slight and the computational cost of training becomes greater than maximum entropy.
3) Feature Combinations
   Various feature combinations are found to be effective for semantic role labeling, especially for machine learning algorithms that inherently treat input features separately. Considering the feature combinations inherently increases the computational cost while training. However, [22] showed that comparable performance could be achieved by combining the linguistically informed features and the Maximum Entropy Approach. Besides, they also found the following combinations for argument identification are useful: predicate-phrase type combination, and predicate-head word combination, compared to just considering the features of path, headword, headword part-of-speech, the distance between constituent and predicate, as well as the predicate specified individually. However, the SRL task has disparities compared to the general multi-classification problems in machine learning in 2 aspects: 1)Bidirectional Relationship; and 2)Highly Imbalance.
   To capture the aspects of syntax and lexical semantics and find the best understanding of the entire background sentence, the problem extends beyond individual prediction per word to consider the bidirectional context, i.e., the tokens around the predicted target.
   There is a severe imbalance between positive samples(words that are arguments for the predicate) and negative samples(background words without any role labels). Normally, machine learning algorithms do not handle such extremely imbalanced cases. To solve

Role	Description	Examples
Agent	Initiator of action, capable of volition	**The batter** smashed **the pitch** into left field. The pilot landed the plane as lightly as a feather.
Patient	Affected by action, undergoes change of state	David trimmed **his beard**. John broke **the window**.
Theme	Entity moving, or being "located"	Paola threw **the Frisbee**. **The picture** hangs above the fireplace.
Experiencer	Perceives action but not in control	**He** tasted the delicate flavor of the baby lettuce. **Chris** noticed the cat slip through the partially open door.
Beneficiary	For whose benefit action is performed	He sliced **me** a large chunk of prime rib, and I could hardly wait to sit down to start in on it. The Smiths rented an apartment **for their son**.
Instrument	Intermediary/means used to perform an action	He shot the wounded buffalo with **a rifle**. The surgeon performed the incision with **a scalpel**.
Location	Place of object or action	There are some real monsters hiding **in the anxiety closet**. The band played **on the stage**.
Source	Starting point	The jet took off **from Nairobi**. We heard the rumor **from a friend**.
Goal	Ending point	The ball rolled **to the other end of the hall**. Laura lectured **to the class**.

TABLE I
SEMANTIC ROLES

this problem, [17] proposed a pipeline of argument identification followed by argument classification. The goal of the first filtering stage is to decide whether a constituent is an argument. The second stage of a multi-class classifier is trained to decide the semantic role for this argument. Similarly, [21] proposed to use support vector for classification with a 2-stage implementation. Besides, [22] proposed an approach is to use a set of heuristics to prune out the majority of the negative samples, for the simple observation that a predicate's roles are generally found in a limited number of syntactic relations to the predicate itself. Some SRL systems combine both of the above approaches.

III. APPROACH

A. Problem statement

To identify the role labels of words automatically, we build a hybrid BERT model leveraging the NomBank-based semantic role labels and the uncased pre-trained BERT.

The NomBank annotation[1] is based on the Penn TreeBank II corpus. We defined our task as classifying the cardinal word-wised role labels for each common noun into ARG0, ARG1, ARG2, Support, and predicate because they occur most frequently with most research attention. Besides, those popular tags provide heuristical insights for the rest of semantic labels of interest.

The typical way to do the Semantic Role Labeling task is to separate it into two stages: argument identification and argument classification. The first stage is to identify and label the words as arguments or non-argument. The second stage is to identify which role the words belong to. Leveraging the power of deep hierarchical models, we can safely simplify these two stages into one multi-classification problem with unbalanced data. To be more specific, our task is to identify 5 role labels of ARG0, ARG1, ARG2, Support, and predicate for each word in the corpus, but some words are background words. Thus, we rephrase our problem into identifying 6 role labels of ARG0, ARG1, ARG2, Support, predicate, and None, where the label None presents the words as non-argument.

More detailed definitions of the argument are shown as follows:

PRED the predicate of the relation

SUPPORT the connecting words

ARG0 Owner Example:

ARG1 Possibilities

- List of (i) set members or (ii) part of PRED
- Description of a property of that set

ARG2 Value and Group

- SHARE nouns: fraction or amoun of share
- Group nouns: Employer, leader or other pivotal entity

Examples

- Mary/ARG0 's portion/PRED of the pizza/ARG1
- the 11 components/PRED to the index/ARG1
- The price/ARG1 rose/SUPPORT 5 %/PRED
- team of managers/ARG1
- team of editors/ARG1
- management/ARG1 team
- editing/ARG1 team
- the largest/ARG2 chunk/PRED of Western Union/ARG1
- California/ARG2 's assembly/PRED

B. Prepossessing

1) Subword Tokenization: Tokenization plays an essential role in NLP as it helps convert the text to numbers which deep learning models can use for processing. The unstructured text data generated by humans should be unified into fixed-size tensors, thus we need to separate the sentences into tokens. Next, we need to map the tokens into their corresponding embedding representations of their semantic information. However, we cannot build the word embedding systems based on our relatively small corpus from scratch. Instead, we use word embedding integrated with BERT. Thus, we could not garentee that each word in our own corpus could find their corresponding mapping in the predefined word embedding systems, especially long words and words with low frequency, and we call them OOV (Out Of Vocabulary). To solve this problem, once OOV occurs, they will be split into their known roots. For example, if the word "dataset" is identified as unknown in the embedding pipeline, then it will be split as "data", "##set". Double hashtags are commonly used to denote subword tokens. The subword tokenization schema provides a straightforward and powerful way to identify the semantic

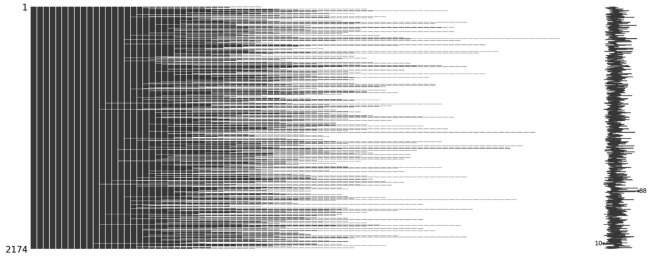

Figure 1. Truncation

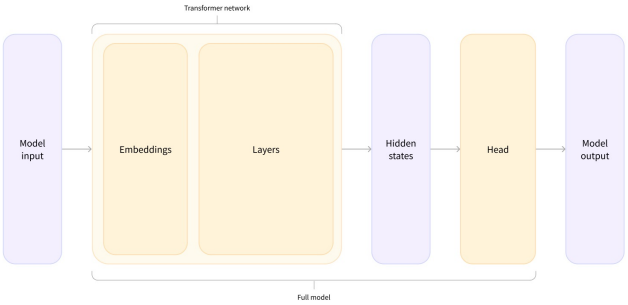

Figure 2. Pipeline

information from OOVs. Besides, it helps to keep the vocab size to a minimum.

Since POS(part of speech) and BIO(Begin/Inside/Other Chunk tag) are categorical variables, we use one-hot vectors as representations. For our specific purpose of concatenating POS and BIO tags as one-hot vectors into the final linear layer, we keep the length and positional information of POS and BIO tags the same as tokenized output. For example, if the OOV "dataset" is with POS tag of "NN" and BIO tag of "B-NP", then its token is ["data", "##set"], and its POS and BIO labels are ["NN", None], ["B-NP", None] respectively.

Algorithm 1 Align Labels with Tokens

```
for iteration = 1, 2, ... do
    if wordID is OOV then
        tokenLabel = noneLabel;
        POSLabel = NonePOS;
        BIOLabel = NoneBIO;
    else
        if now it's a start of a new word then
            update current word to wordID
        end if
        tokenLabel = labels[wordID];
        POSLabel = POSs[wordID];
        BIOLabel = BIOs[wordID];
    end if
end for
```

Algorithm 2 Tokenize and Align Labels

```
Assign "output" with the tokenized result calculated by the pre-trained BERT tokenizer with truncation and padding;
for iteration = 1, 2, ... do
    wordID = i-th tokenized result
    Calculate [token, POS, BIO] from Align function with wordID;
    add token to tokenLabels Array;
    add POS to POSLabels Array;
    add BIO to BIOLabels Array;
end for
add tokenLabels, POSlabels and BIOLabels to "output"
return output
```

2) Truncation with max length: The number of tokens in sentences varies. We sampled and visualized the distribution of token length as Figure 1. As we can see in the 2174 samples, the maximum number of tokens in a sentence is 88, while the minimum length is 10. To boost our computational efficiency, we constraint the maximum number of tokens to 48 to significantly reduce the size of the input matrix while keeping most of the information since most of the tokens after the 48th are empty.

3) Padding: On the other hand, sentences are also pretty likely to be with less than 48 tokens. To train the model batch by batch rather than sentence by sentence, it's vital to pad the shorter sentences into the same length of 48. Besides, we use attention masks to identify whether the tokens should be assigned labels in the output layer or they are just padding tokens; thus no need to make actual predictions.

C. BERT

Following the frontier research of natural language processing, we use Transformer as our basic network hierarchy for semantic information extraction. However, the plain BERT model does not enjoy the merits of classic linguistic contributions derived from the traditional statistical learning perspective. Thus, we make efforts to augment the plain BERT model with classic linguistic labels of POS and BIO for our specific focus on inherent semantic relationship identification and extraction in token granularity.

Besides, the token and sentence number are excluded from our selected feature because our research interest focuses on how those traditional linguistic labels help in nowadays's a deep hierarchical black box. Before fine-tuning BERT, we need to shuffle the sentences to prevent over-fitting by ensuring the order of sentences is irrelevant to the semantic role label itself. Additionally, the token number is only used for reconstructing the prediction to the initially given order for calculating the metrics of model performance.

Following the frontier research of natural language processing, we use Transformer as Figure 2 to be our basic network hierarchy for semantic information extraction. However, the plain BERT model does not enjoy the merits of classic linguistic contributions derived from the traditional statistical learning perspective. Thus, we make efforts to augment the plain BERT model with classic linguistic labels

of POS and BIO for our specific focus on inherent semantic relationship identification and extraction in token granularity.

Besides, the token and sentence number are excluded from our selected feature because our research interest focuses on how those traditional linguistic labels help in nowadays's deep hierarchical black box. Before fine-tuning BERT, we need to shuffle the sentences to prevent over-fitting by ensuring the order of sentences is irrelevant to the semantic role label itself. Additionally, the token number is only used for reconstructing the prediction to the originally given order for calculating the metrics of model performance.

D. Multi-class linear classifier

The batch size of our training is 64, the number of tokens of each sentence was unified to 48 by truncation and padding with masking. Thus, the BERT delivers tensors with dimensions of (64,48,768), where 768 is the output and embedding representation extracted by our fine-tuned BERT model.

To merge the linguistic labels created by humans to our BERT to build a hybrid deep system, we concatenated the one-hot vectors representing the categorical information of POS and BIO in the third dimension of the output of BERT. The dimension of concatenated tensor became (64, 48, 866).

Next, we could rephrase our task in the final stage to be a logistic regression, i.e., the header of our hybrid model. A linear layer helps to classify the $X \in R^{866}$ dimensional input $\{X_i\}$ for $i \in \{1, 2, \cdots, n\}$, where n is 64×8 for each batch.

To update the parameters in the final linear layer for multi-classification, we use the Cross Entropy with softmax function. The softmax function gives us a vector $\hat{y}$, which we can interpret as estimated conditional probabilities of each label $\boldsymbol{y_j}$, for $j \in \{0, 1, 2, \cdots, 5\}$ representing ARG0, ARG1, ARG2, Predicate, Support and Background words, respectively.

Thus, we can compare the estimates with true labels by checking the Likelihood defined as:

$$P(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{i=1}^{n} P(\boldsymbol{y_i}|\boldsymbol{x_i}) \tag{1}$$

The reason why we use factorization is that we assume each label is drawn independently from its respective distribution $P(y|x_i)$.

Our goal is to maximize the Likelihood. To reduce the computational cost, we could take the negative logarithm to transform the maximum likelihood problem into the equivalent problem of minimizing the negative log-likelihood:

$$-logP(\boldsymbol{Y}|\boldsymbol{X}) = \sum_{i=1}^{n} -logP(\boldsymbol{y_i}|\boldsymbol{x_i}) ::= \sum_{i=1}^{n} l(\boldsymbol{y_i}, \hat{\boldsymbol{y}}_{\boldsymbol{i}}) \tag{2}$$

, where for any pair of label **y** and model prediction over our 6 classes, the loss function $\boldsymbol{l}$ is

$$l(\boldsymbol{y}, \hat{\boldsymbol{y}} = -\sum_{j=1}^{m} (y_j) log \hat{\boldsymbol{y}}_{\boldsymbol{j}} \tag{3}$$

, where m = 6, e.g. the number of labels. In the he loss function, $\boldsymbol{y_j}$, for $j \in \{0, 1, 2, \cdots, 5\}$ is either 1 or 0. For example, if $y_i = [0, 1, 0, 0, 0, 0]$ and $\hat{y_i} = [1, 0, 0, 0, 0, 0]$, it means that the true label of corresponding X_i^{866} is ARG1, but the system predict X_i as ARG0. It's straightforward that now $l(\boldsymbol{y}, \hat{\boldsymbol{y}} = -\sum_{j=0}^{5} (y_j) log \hat{\boldsymbol{y}}_{\boldsymbol{j}} = 0$, because if $\boldsymbol{y_j}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{j}}$ is not equal in position k, $k \in 0, 1, 2, 3, 4, 5$, then $\boldsymbol{y_j} \times \boldsymbol{haty_j}$ = 0. $l(\boldsymbol{y}, \hat{\boldsymbol{y}}$ is 1 if and only if $\boldsymbol{y_j}$ and $\hat{\boldsymbol{y}}_{\boldsymbol{j}}$ is equal.

Now it's time to calculate $\hat{\boldsymbol{y}}$. We use Softmax function:

$$\hat{\boldsymbol{y}} = \frac{exp(o_j)}{\sum_{k=1}^{q} exp(o_k)} \tag{4}$$

Thus, the loss function becomes:

$$\begin{aligned} l(\boldsymbol{y_j}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}) &= -\sum_{j=1}^{m} y_j log \hat{\boldsymbol{y}} \\ &= \sum_{j=1}^{m} y_j log \frac{exp(o_j)}{\sum_{k=1}^{m} exp(o_k)} \\ &= \sum_{j=1}^{m} y_j log \sum_{k=1}^{m} exp(o_k) - \sum_{j=1}^{q} y_j o_j \\ &= log \sum_{k=1}^{m} exp(o_k) - \sum_{j=1}^{m} y_j o_j \end{aligned} \tag{5}$$

, where o_j is the original numeric output with m dimensions and needs to be normalized to the [0,1] probability. As for the derivative with respect to any logit o_j, we have:

$$\partial_{oj} l(\boldsymbol{y_j}, \hat{\boldsymbol{y}}_{\boldsymbol{j}}) = \frac{exp(o_j)}{\sum_{k=1}^{q} exp(o_k)} - y_j = softmax(o)_j - y_j. \tag{6}$$

IV. Experiment

Dataset. We use two datasets for a comprehensive evaluation: (1) % Norm Bank, (2) All Norm Bank.

Metrics. On each multi-class, we employ accuracy, precision, recall, and F1_Score separately for quantitative analyses. Additionally, we determined the mean metrics for five various labels. where TP, TN, FN represents the true positive, true negative and false negatives for each token respectively. Also, P and N represent the overall positive and negatives

Baselines. We compare our method(BERT+POS+BIO) with three baselines including: (1) Vanilla BERT [5] (2) BERT+POS, and (3) BERT+BIO. The detailed ablation studies are provided in Tab.II and Tab.III.

Overall comparisons. Tab.II demonstrates that our approach exceeds all other baselines almost in every category. The categorization result is only marginally enhanced by the human-labeled data, though. The precision would similarly decrease by around 7% with BIO tags included. Similarly, if we did the same ablation on all_Norm Bank dataset, the F1 score of our model outperforms other baselines regardless that there is a slight variation in precision and recall.

Detailed comparisons. To check how the model trained would impact all label classifications, we list all metrics for

TABLE II

QUANTITATIVE ANALYSIS ON THE % AND ALL NORM BANK DATASETS. THE QUALITY OF THE MULTI-CLASS CLASSIFICATION IS MEASURED BY F1 SCORE, PRECISION, RECALL, AND ACCURACY. TWO DATASETS' MEAN METRICS ARE PROVIDED. THE TRAIN-TEST SPLIT HAS BEEN APPLIED AND THIS RESULT IS ONLY TRAINED WITH 3 EPOCHS.

Dataset	% Norm Bank				All Norm Bank			
Metrics	F1	Precision	Recall	Accuracy	F1	Precision	Recall	Accuracy
Vanilla BERT	55.02%	67.36%	54.39%	81.52%	19.74%	**51.91%**	12.51%	43.80%
BERT+POS	61.11%	70.58%	61.17%	81.72%	19.87%	50.44%	12.69%	43.94%
BERT+BIO	58.27%	**77.79%**	58.85%	81.64%	20.37%	51.32%	**12.95%**	43.97%
BERT+POS+BIO	**66.68%**	70.43%	**65.80%**	**81.80%**	**20.45%**	49.81%	12.84%	**43.98%**

TABLE III

DETAILED QUANTITATIVE ANALYSIS ON THE % AND ALL NORM BANK DATASETS. THE METRICS ARE AS SAME AS IN TAB.II. INSTEAD, METRICS FOR DIFFERENT LABELS ARE LISTED.

Dataset	Label	% Norm Bank			All Norm Bank		
Metrics		F1	Precision	Recall	F1	Precision	Recall
Vanilla BERT	ARG0	N/A	N/A	N/A	22.01%	56.99%	14.20%
	ARG1	18.75%	10.34%	100%	17.81%	47.51%	11.19%
	ARG2	N/A	N/A	N/A	12.60%	35.40%	7.87%
	PRED	83.56%	71.76%	100%	26.29%	64.04%	16.64%
	SUPPORT	62.75%	79.99%	51.66%	20.01%	55.64%	12.65%
BERT+POS	ARG0	N/A	N/A	N/A	21.45%	52.89%	13.86%
	ARG1	33.76%	68.42%	22.41%	17.52%	47.88%	11.01%
	ARG2	N/A	N/A	N/A	13.07%	33.64%	8.35%
	PRED	82.19%	70.59%	98.36%	28.18%	60.96%	18.43%
	SUPPORT	67.37%	72.73%	62.75%	19.16%	56.84%	11.82%
BERT+BIO	ARG0	N/A	N/A	N/A	21.45%	55.37%	13.61%
	ARG1	23.88%	88.89%	13.79%	18.56%	48.21%	11.73%
	ARG2	N/A	N/A	N/A	14.48%	37.18%	9.21%
	PRED	83.56%	71.76%	100%	26.53%	62.40%	16.97%
	SUPPORT	67.37%	72.73%	62.75%	20.86%	53.46%	13.21%
BERT+POS+BIO	ARG0	N/A	N/A	N/A	22.55%	54.17%	12.93%
	ARG1	46.32%	59.46%	37.93%	17.16%	47.63%	10.73%
	ARG2	N/A	N/A	N/A	15.01%	34.19%	9.91%
	PRED	84.89%	75.64%	96.72%	28.23%	59.39%	18.62%
	SUPPORT	68.82%	76.19%	62.75%	19.30%	53.67%	12.02%

each label. Note that % Norm Bank does not include ARG0 and ARG2 classes in the dataset. It turns out that our model performs better when classifying predicate and it performs the worst when classifying ARG2. The potential explanation is that the amount of training dataset for ARG2 is far less than other categories.

V. DISCUSSION

Our attempt to use the feature combinations to improve the SRL performance with deep approaches did gain improvement in performance though it's slight.

Theoretically, it's vital to ensure that the feature combinations are robust enough for training data, and they should be informative of the semantic role. One crucial prepossessing is to integrate all of the role labels into one unique sentence because it's universal for the original Lexical Resources to make some copies of the same sentences to indicate different words were assigned the same label.

In practice, the empirical process of building SRL system requires reasonable and understandable feature combinations linguistically, but it does not always result in improved performances. It must be weighed against machine learning matrics in the training process, where a suitable and well-designed customized metric in the learning process is necessary. As for the data distributions, sparsity and imbalance is also the key to building a usable SRL system.

REFERENCES

[1] Adam Meyers, Ruth Reeves, Catherine Macleod, Rachel Szekely, Veronika Zielinska, Brian Young, and Ralph Grishman. The NomBank Project: An Interim Report. In *Proceedings of the Workshop Frontiers in Corpus Annotation at HLT-NAACL 2004*, pages 24–31, Boston, Massachusetts, USA, May 2004. Association for Computational Linguistics.

[2] Atro Voutilainen. *Part-of-speech tagging*, volume 219. The Oxford handbook of computational linguistics, 2003.

[3] Erik F Sang and Jorn Veenstra. Representing text chunks. *arXiv preprint cs/9907006*, 1999.

[4] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia. Association for Computational Linguistics*, 1(3):328–339, 2018.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] Ming Zhou, Nan Duan, Shujie Liu, and Heung-Yeung Shum. Progress in neural nlp: modeling, learning, and reasoning. *Engineering*, 6(3):275–290, 2020.

[7] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

[8] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[10] 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*, pages 1631–1642, 2013.

[11] Michael I Jordan, Michael J Kearns, and Sara A Solla. *Advances in Neural Information Processing Systems 10: Proceedings of the 1997 Conference*, volume 10. Mit Press, 1998.

[12] Martha Palmer, Daniel Gildea, and Nianwen Xue. Semantic Role Labeling. *Synthesis Lectures on Human Language Technologies*, 3(1):1–103, January 2010. Publisher: Morgan & Claypool Publishers.

[13] Collin F Baker, Charles J Fillmore, and John B Lowe. The berkeley framenet project. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, 1998.

[14] Karin Kipper Schuler. *VerbNet: A broad-coverage, comprehensive verb lexicon*. University of Pennsylvania, 2005.

[15] Paul R Kingsbury and Martha Palmer. From treebank to propbank. In *LREC*, pages 1989–1993, 2002.

[16] Xiaoshi Zhong and Erik Cambria. Time expression recognition using a constituent-based tagging scheme. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 983–992, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.

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

[18] Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. Using predicate-argument structures for information extraction. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 8–15, 2003.

[19] Michael Fleischman, Namhee Kwon, and Eduard Hovy. Maximum entropy models for framenet classification. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 49–56, 2003.

[20] Kristina Toutanova, Aria Haghighi, and Christopher D Manning. Joint learning improves semantic role labeling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 589–596, 2005.

[21] Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H Martin, and Daniel Jurafsky. Support vector learning for semantic argument classification. *Machine Learning*, 60(1):11–39, 2005.

[22] Nianwen Xue and Martha Palmer. Calibrating features for semantic role labeling. In *EMNLP*, pages 88–94, 2004.