# Systems at SDU-2021 Task 1: Transformers for Sentence Level Sequence Label

**Feng Li, Zhensheng Mai, Wuhe Zou, Wenjie Ou, Xiaolei Qin, Yue Lin, Weidong Zhang***
Netease Games AI Lab
{lifeng06, maizhensheng, zouwuhe, ouwenjie, qinxiaolei, gzlinyue, zhangweidong02}@corp.netease.com

**Abstract**

This paper describes the system proposed for addressing the research problem posed in Task1 of scientific document understanding (SDU@AAAI-2021): Acronym Identification. We proposed an end-to-end model that takes the text as input and corresponding to each word gives the label of word to be acronyms (short-forms) or their meanings (long-forms). We take experiment on several totally different ideas, including features engineering, transformer model, multi-task learning, Span and CRF. Our result shows that feature-based method can handle this task well, and transformer-based models are particularly effective in this task. Moreover, different model frameworks complement each other. We achieved the best f1 score of 0.931 on test dataset and were ranked second.

## Introduction

An obstacle of scientific document understanding (SDU) is the extensive use of acronyms which are shortened forms of long technical phrases (Veyseh et al. 2020a,b). In order to correctly process a document, an SDU system should be able to identify acronyms and their correct meanings. As acronyms might be defined either locally in the same document or globally in an external dictionary with multiple meanings, the challenge is to capture both local definitions and disambiguate acronyms which are not defined in documents. The SDU task aims at solving this problem. The dataset provided for this task is in English, and task description GitHub [1] repository provided by the organizers describes the task, data, evaluation, rule-based baseline model (Schwartz and Hearst 2003) and its result.

We tried four different approaches for the task. Our feature-based approach use BiLSTM-CRF (Luo et al. 2018) framework. For the inputs to BiLSTM-CRF, we concatenate the output of Roberta (Liu et al. 2019), GloVe (Pennington, Socher, and Manning 2014) vectors and the task-specific features.

Our XLNet-Softmax approach is inspired by the Emphasis Selection paper (Shirani et al. 2020, 2019; Singhal et al. 2020). This approach involves transfer learning using Transformer based models. It is a transformer-based model with the BiLSTM layer, the attention layer (Bahdanau, Cho, and Bengio 2014), and fully connected layer on top. The small size of the dataset was a bottleneck that can be countered by using transfer learning via the pre-trained models. So we used XLNet as transformer-based models. Motivated by MRC for NER task (Li et al. 2019), we apply BERT-Span approach to predict the start and end positioins of spans. BERT-CRF (Souza, Nogueira, and Lotufo 2019) approach is employed to caputre the transfer information betweent different tags. Our feature-based approach independently achieved f1 score of 0.9281 on test data. Our Bert-CRF model, Bert-Span model and XLNet-SoftMax model are ensembled to achieve f1 score of 0.931 on test data. As the Bert-Span model predict probability of label's begin and end. XLNet-SoftMax and BERT-CRF directly predict probability of different labels. It is not easy to ensemble the result together. In the end, we tried ensemble of these three models by voting on the label-level. Our code is available on https://github.com/NetEase-GameAI/sdu_task1.

## Background

### Problem definition

We take this task as sentence level sequence labeling problem. Given a sequence of words or tokens in a text, the task is to compute a label $L_i$ for each $x_i$ which indicates the boundaries of short-form (i.e., acronym) and long-form (i.e., phrase).

### Evaluation Metric

The approaches are evaluated based on their macro-averaged precision, recall, and F1 scores on the test set computed for correct predictions of short-form (i.e., acronym) and long-form (i.e., phrase) boundaries in the sentences. A short-form or long-form boundary prediction is counted as correct if the beginning and the end of the predicted short-form or long-from boundaries equal to the ground-truth beginning and end of the short-form or long-form boundary, respectively. The official score is the macro average of short-form and long-form prediction F1 score.

* Corresponding author


[1] https://github.com/amirveyseh/AAAI-21-SDU-shared-task-1-AI

## Data

The dataset consisting of 20000+ sentences extracted from English scientific papers published at arXiv. The dataset is divided into training (17506), development (1715) and test (1750) sets by organizers. The training and development datasets are manually labeled. Each sample has three attributes: tokens, labels and id. The short-form and long-form labels of words are in BIO format. The labels B-short and B-long identifies the beginning of a short-form and long-form phrase, respectively. The labels I-short and I-long indicates the words inside the short-form or long-form phrases. Finally, the label O shows the word is not part of any short-form or long-form phrase.

## Baseline models

The baseline method proposed by Schwartz is a rule-based method (Schwartz and Hearst 2003) (Character Match). To identify acronyms, if more than 60% of the characters of a word are uppercased, this model recognizes it as acronym (i.e., short-form). To identify the long-form, it compares the characters of the acronym with the characters of the words that are before or after the acronym up to a certain window size. If the characters of these words could form the acronym, they are labeled as long-form. The official scores for this baseline are: Precision: 93.22%, Recall: 78.90%, F1: 85.46%.

BiLSTM-CRF (Liu et al. 2019) and BERT-large model (Devlin et al. 2019) are the another baseline models, since the task is a standard sequence labeling task.

# System Overview

In this section, we introduce the four types of approaches in detail, and the overall architecture of our approaches is shown in Figure 1.

## Features

Since the data is relatively regular, i.e., a word with more than 60% of the characters in uppercase are most likely to be a acronym, and its long-form may probably appear before or after it up to a certain window size[2], we consider to extract some task-specific features to improve our models. After exploring the whole dataset, we extract the following features and concatenate them as additional inputs.

- **Token Length** Token length is a commonly used feature in NLP. In our experiments, we statistic the length of every tokens and decide to group them by their lengths, except for the tokens whose lengths are greater than 15, in which we set them a separate group. In this way, we have a 16-dimensional feature.
- **POS Tagging** Part-of-speech Tagging is another commonly used feature in NLP. We first count the number of categories of POS tags on the entire dataset, then use N-dimensional one-hot features to represent. In our experiments, N is 43.
- **Letter Case** According to our data analysis, a word in uppercase has an 85.4% probability of being acronym. So we design a 4-dimensional one-hot feature to represent that if the words are uppercase, lowercase, capitalization or others.
- **Token Type** Generally speaking, a token composed entirely of punctuation cannot be an acronym or a long-form, but a token that is alphanumeric strings or a mixture of English letters and punctuation is most likely to be an acronym or long-form. So we design another 4-dimensional one-hot feature to represent that if the tokens are just all punctuation, all English strings, mixture strings, or strings with non-English.
- **Whether Begin with Digit and End with English** A prior knowledge is that a token matching this pattern is probably a acronym, such as 2D, 3D and 5G. A 2-dimensional one-hot feature about this is extracted.
- **Character Match Features** We notice that the official baseline, i.e. **Character Match**, can achieve a very high precision scores, 93.22, which may be used to design some strong features. In our experiments, we use the predictions of **Character Match** and group the tokens into their predicted labels ('O', 'B-long', 'I-long', 'B-short', 'I-short') to present a 5-dimensional one-hot feature.
- **Previous Tokens** During our data exploring, we find that acronyms or long-forms are often following some specific tokens, such as *'the', 'a', 'called', 'mean', 'of', 'or', 'and', '(', ')', '-', ',', ':', '"',* . So we design a 14-dimensional one-hot feature to represent it.
- **Next Tokens** The acronyms and long-forms are also often followed by the same specific tokens as **Previous Tokens**. So another 14-dimensional one-hot feature is extracted.
- **Pseudo Map Short to Long** We design two types of features according to the facts that acronyms are the short forms of longer phrases. Firstly we design a pattern that can match most of acronyms. This pattern must consist of letters and digits and contains at least two letters in uppercase, except for composing of all digits. Tokens which match the pattern are collected as a set, named *shorted set*, while the letters in these tokens are collected as another set, named *shorted letter set*. The first type of features is to determine if each token matches the pattern, or the first letter is in the *shorted letter set*, or other cases. So a 3-dimensional one-hot feature can be extracted. The second type of features is to traverse the sentence to see if tokens matches the pattern or the first letter is matching part of acronym with label 'B' or 'I'. For example, token 'mean' can be the beginning of the acronym **MSE**, which means 'mean square error', and the inside of the acronym **RMSE**, which means 'root mean square error'. There can be five cases in this way, that a token may in the shorted set meaning that it is acronym itself, or be the beginning or the inside (appearing in only one acronym), or both the beginning and the inside (appearing in multiple acronyms), or neither the beginning nor the middle (not appearing in any acronym). Then a 5-dimensional feature is extracted.

[2]https://github.com/amirveyseh/AAAI-21-SDU-shared-task-1-AI

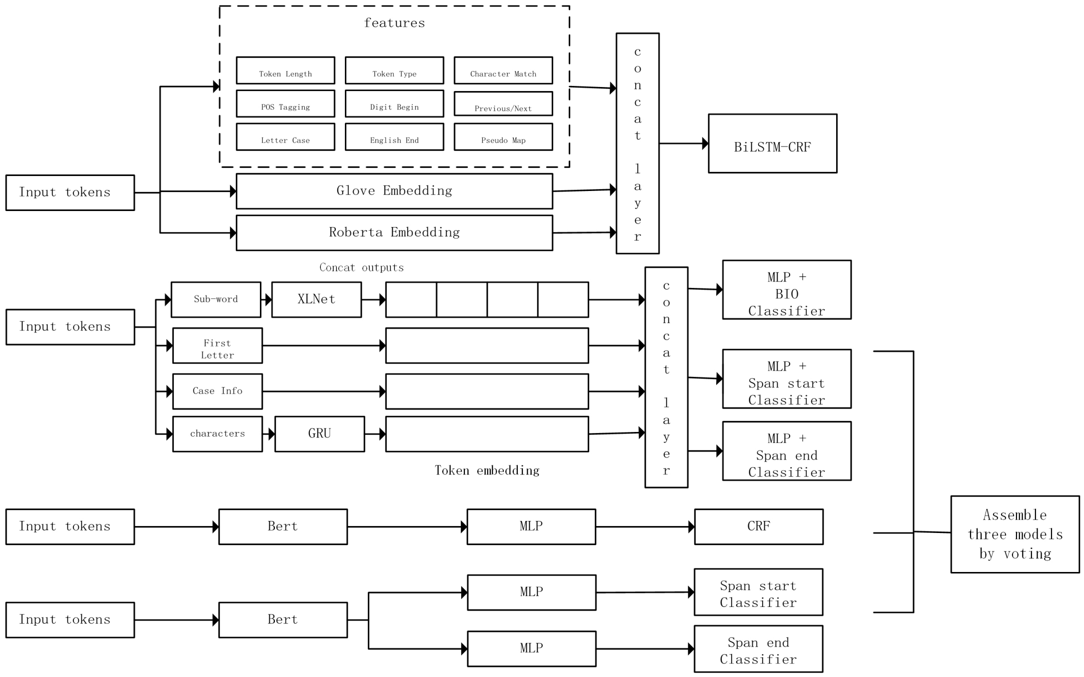


Figure 1: Models for approaches

We concatenate the features mentioned above all. Additionally, since the GloVe vectors[3] are usually used as supplementary input for the bert-based models, we also concatenate the GloVe vector with our handful features to generate a 410-dimensional feature.

## XLNet-Softmax

In this approach, words are tokenized into sub-words using corresponding tokenizer of transformer model. For the Transformer-base models, we used pre-trained XLNET (Yang et al. 2019). Sub-word embedding is obtained by concatenating the hidden layers of all encoder layers of the XLNet model. Token embedding is obtained by average all embedding of the token's sub-words. To the nature of the problem, first letter of token is important. We added embedding of first letter (no case) and tag indicating whether first letter is capital. The token was also encoded with GRU on its letters. And the letters and first letter share the same embedding layer. All these token embedding are concatenated. On one hand, the token embedding is passed through a fully connected layer with a dropout layer and a soft-max layer, which gives the probabilistic score of BIO label for each token. BIO cross entropy loss is calculated with the score and ground truth label. The labels consist of B-short, I-short, B-long, I-long and Others. On the other hand, the token embedding is passed through other fully connected layer with a dropout layer and a soft-max layer, which give the segment probability of each token. Cross entropy loss for Short-Phrase-Segmentation is calculated with the probability and ground truth segmentation. The segmentation consists of short-start, short-end. Simultaneously, the cross entropy loss for Long-Phrase-Segmentation is calculated. Finally, we sum the three parts of loss. Addition to the multi-task work frame, we tried adversarial Learning, which is inspired by MRC framework (Li et al. 2019).

## BERT-CRF approach

We design a BERT-CRF model (Devlin et al. 2019; Lafferty, McCallum, and Pereira 2001) for the sequence labeling task. The model architecture is composed of a BERT model with a token-level classifier on top followed by a Linear-Chain CRF. For an input sequence of tokens, BERT outputs an encoded token sequence, and the classification model projects each token's encoded representation to the tag space. The output scores of the classification model are then fed to the CRF layer, whose parameters are a matrix of tag transitions, whose element is represents the score of transitioning from one tag to other tag. The matrix includes 2 additional states: start and end of sequence. We compute predictions and losses only for the first sub-token of each token.

## BERT-Span approach

In addition to treat the task as a sequence labeling problem, we also model it as determining the boundary of phrase spans, including short-forms and long-forms.

We employ two binary classifiers to output multiple start indexes and multiple end indexes, one to predict whether each token is the start index or not, the other to predict whether each token is the end index or not. Given the representation matrix output from BERT encoder, the model predicts the probabilities of each token being the start or end position of need forms. At training time, each sentence is

[3]https://nlp.stanford.edu/projects/glove/

| Model | validation f_score | test f_score |
|---|---|---|
| Character Match | 0.8546 | – |
| BiLSTM-CRF | 0.8439 | – |
| BERT-large | 0.9102 | 0.9065 |
| 1. Fixed-RoBERTa+Feature+BiLSTM-CRF (K-fold) | 0.9255 | 0.9281 |
| 2. XLNet-SoftMax | 0.9345 | 0.9188 |
| 3. XLNet-SoftMax+Multi-Task+Adversarial-Learning | 0.9371 | 0.9216 |
| 4. BERT-CRF | 0.9276 | 0.9082 |
| 5. BERT-Span | 0.9334 | 0.9116 |
| 6. Model vote (Model-2,4,5) | – | 0.9311 |
| 7. BERT-Span+BERT-CRF | 0.9411 | – |

Table 1: The results of different models on development and test datasets

paired with two label sequences $Y_{start}$ and $Y_{end}$ of length $n$ representing the ground-truth label of each token $x_i$ being the start index or end index of any form. The loss of two weighted cross-entry losses are jointly trained in an end-to-end fashion with parameters shared at the BERT layer. At test time, we select span based the same tag from the start indexes predictions $\hat{I}_{start}$ and end indexes predictions $\hat{I}_{end}$.

### Multi-task with BERT-Span and BERT-CRF

In order to take advantage of the above two models, we also employ multi-task model based on BERT-CRF model and BERT-Span model with the same BERT layers. And we train the multi-task model using BERT-CRF loss and BERT-Span loss weighted sum, and in prediction process, we decode the tags with the BERT-Span model. And the multi-task learning model touchs the highest f-score with a single model on the development dataset, but we have no enough time to verify the effect on the test dataset.

## Experiment Setup

Our implementation uses PyTorch [4] library for deep learning models and the Transformers [5] library by Hugging face for the pre-trained transformer models and their tokenizers.

In XLNet Softmax model, the pre-trained XLNet-big-cased model was used without freezing the layers, and the outputs of all the layers were concatenated. Additionally, letters embedding, first letter embedding and tag indicating whether capitalize the first letter were concatenated. We use hidden dimension of 128 and pre-trained large model of XLNET (large-xlnet-cased). The tokens size is padding to 345 and letter size is padding to 98. For the classifier, two fully connected layers are used with ReLU activation function. Dropout layers with a probability of 0.3 are added to avoid overfitting. Cross-Entropy loss is used for training phase. F1 score is used as performance metrics for validation. Adam optimizer with the learning rate of 2e-5 is used. The model is fine-tuned for 4 epochs. The reported test result is corresponding to the best score on validation set.

In BERT-CRF approach, We initialize the model with bert-large-wwm [6], and initial learning rates are 5e-5 and 5e-2 for BERT and CRF respectively, and the batch size is 16.

In BERT-Span approach, We initialize the model with bert-large-wwm, and initial learning rates are 5e-5, and the batch size is 16.

Our models are trained on 8 Nvidia Tesla P40 cards, and predicted on one GPU card. Take BERT-Span model as an example, average train time of one epoch is 27 minutes and average predict time is 45 seconds for development dataset.

## Results

The main result of our models and some baseline is shown in Table 1. Through the Table 1 shown, the pretrained model has more advantages and reaches F1: 90.65% on test dataset, since the train dataset is relatively small. Voting on the results of XLNet-Fostmax, BERT-CRF, BERT-Span reaches F1:93.11% on test dataset. In fact, multi-task of BERT-CRF and BEET-Span gets the highest F1:94.11% on development dataset, but we don't have enough time to verify on test dataset.

From the results of BERT-CRF and BERT-Span, it is relatively earier to predict the begin and end of short-forms or long-forms. Moreover, based on result of multi-task model of BERT-CRF and BERT-Span, multi-task learning can learn the advantages of the two models.

We attempted numerous small changes to our models. This included some specific attempts particular to each approach and some ablation experiments. For the XLNet-Softmax, we compared cased sub-word and no cased sub-word. We found that cased sub-word is particular important in this task. We tried ablating the multi-task metric and adversarial Learning work frame respectively. We found that multi-task metric and adversarial Learning work frame slightly improve result. We tried ablating the first letter embedding, capital letter tag and GRU encode for each token's letters. We found that first letter embedding is slightly effective. We also tried replace last fully connect layers with LSTM or transformer. We found that fully connect layer is ok. As the submission chance is limited, we did not try all these on test data.

[4] https://pytorch.org/

[5] https://github.com/huggingface/transformers

[6] https://huggingface.co/bert-large-cased-whole-word-masking/tree/main

## Conclusion

We described the systems used for submission in the Task1 of SDU@AAAI-2021. The task was a sentence level sequence label task. We tried several approaches which used Transformer-based models like BERT, RoBERTa, XLNet. These models are pre-trained and hence, perform well on the small dataset after fine-tuning. There was much future work left. Firstly, transformer-based approach and feature-based approach should be well fused. As we have only about one week to study the task, fusion was unfinished. We test transformer-based approach and feature-based approach respectively on test data. The two approaches reach f1 score of 0.931 and 0.9281, ranked second and third according to the leaderboard. Secondly, inside transformer-based approach, we ensemble different models by simply vote. This was primal method and should be improved. Thirdly, after the competition, we further studied the task. Inspired by the BERT-CRF model and BERT-Span model, we proposed a Multi-task model with BERT-Span and BERT-CRF, which employ multi-task model based on BERT-CRF model and BERT-Span model with the same BERT layers, and reach a high f1 score of 0.943 on validation set.

## References


Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Lafferty, J.; McCallum, A.; and Pereira, F. C. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data .

Li, X.; Feng, J.; Meng, Y.; Han, Q.; Wu, F.; and Li, J. 2019. A unified mrc framework for named entity recognition. *arXiv preprint arXiv:1910.11476* .

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* .

Luo, L.; Yang, Z.; Yang, P.; Zhang, Y.; Wang, L.; Lin, H.; and Wang, J. 2018. An attention-based BiLSTM-CRF approach to document-level chemical named entity recognition. *Bioinformatics* 34(8): 1381–1388.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Schwartz, A. S.; and Hearst, M. A. 2003. A Simple Algorithm For Identifying Abbreviation Definitions in Biomedical Text. *Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing* 4(8): 451–462.

Shirani, A.; Dernoncourt, F.; Asente, P.; Lipka, N.; Kim, S.; Echevarria, J.; and Solorio, T. 2019. Learning emphasis selection for written text in visual media from crowd-sourced label distributions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 1167–1172.

Shirani, A.; Dernoncourt, F.; Lipka, N.; Asente, P.; Echevarria, J.; and Solorio, T. 2020. Semeval-2020 task 10: Emphasis selection for written text in visual media. *arXiv preprint arXiv:2008.03274* .

Singhal, V.; Dhull, S.; Agarwal, R.; and Modi, A. 2020. IITK at SemEval-2020 Task 10: Transformers for emphasis selection. *arXiv preprint arXiv:2007.10820* .

Souza, F.; Nogueira, R.; and Lotufo, R. 2019. Portuguese named entity recognition using BERT-CRF. *arXiv preprint arXiv:1909.10649* .

Veyseh, A. P. B.; Dernoncourt, F.; Nguyen, T. H.; Chang, W.; and Celi, L. A. 2020a. Acronym Identification and Disambiguation shared tasksfor Scientific Document Understanding. In *Proceedings of the AAAI-21 Workshop on Scientific Document Understanding*.

Veyseh, A. P. B.; Dernoncourt, F.; Tran, Q. H.; and Nguyen, T. H. 2020b. What Does This Acronym Mean? Introducing a New Dataset for Acronym Identification and Disambiguation. In *Proceedings of the 28th International Conference on Computational Linguistics*, 3285–3301.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R. R.; and Le, Q. V. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, 5753–5763.