1. Introduction

Natural Language Processing (NLP)

Natural language processing, also known as NLP, is a subfield of artificial intelligence (AI), which will focus on allowing computers to understand and process human language.

Natural Language Understanding (NLU)

Language comprehension is a group of techniques in natural language processing that use algorithms to extract the necessary information from input that is sentences presented in text or speech form. Natural Language Understanding (NLU) systems are often used to create bots that support chatting and interacting with users without supervision.

Natural Language Understanding for Smart Home

A smart home system is a convenient home system where devices can be controlled remotely through user commands. From the user's commands, the NLU system will receive and extract the necessary information so that the system can understand the user's request, thereby performing the actions according to their request.

This challenge is about Intent Detection and Slot Filling (IDSF) but utterances come from Vietnamese and are specific to a smart home system. For Intent Detection and Slot Filling (IDSF). We can break it into 2 subproblems, which are Slot tagging and Intent detection. These 2 problems are formulated in 2 very famous tasks, intent classification, and slot tagging.

2. Related works

Transformer

Before Google published the article on Transformers (Attention Is All You Need), most natural language processing tasks, especially machine translation (Machine Translation) used Recurrent Neural Networks (RNNs) architecture. The weakness of this method is that it is difficult to catch the long dependence between words in the sentence and the training speed is slow due to sequential input processing. Transformers were born to solve these two problems; its variants like BERT and GPT-2 generate new state-of-the-art for NLP related tasks.

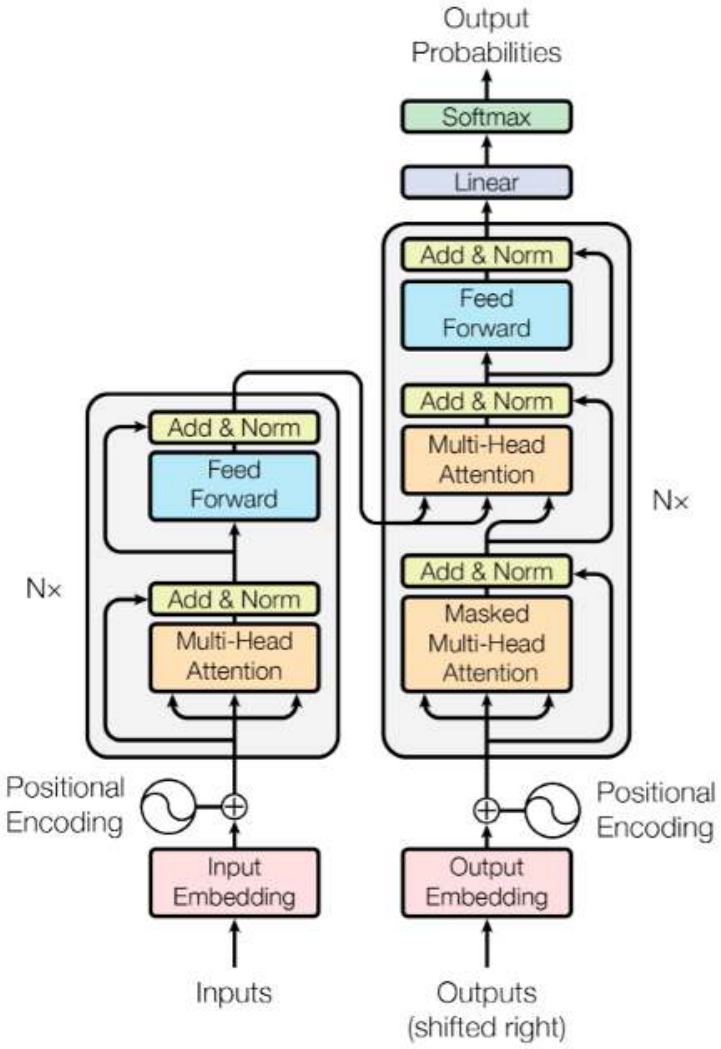


Figure: Transformer architecture

Transformers architecture also uses 2 parts Encoder and Decoder quite similar to RNNs. The difference is that the input is pushed in at the same time. That's right, at the same time; and there will be no timestep concept in Transformers anymore. The Self-Attention mechanism has replaced the "recurrent" of RNNs.

XLM-RoBERTa

XLM-RoBERTa is a multilingual model trained on 100 different languages. It is based on Facebook's RoBERTa model released in 2019. It is a large multi-lingual language model, trained on 2.5TB of filtered Common Crawl data. Unlike some XLM multilingual models, it does not require lang tensors to understand which language is used and should be able to determine the correct language from the input ids.

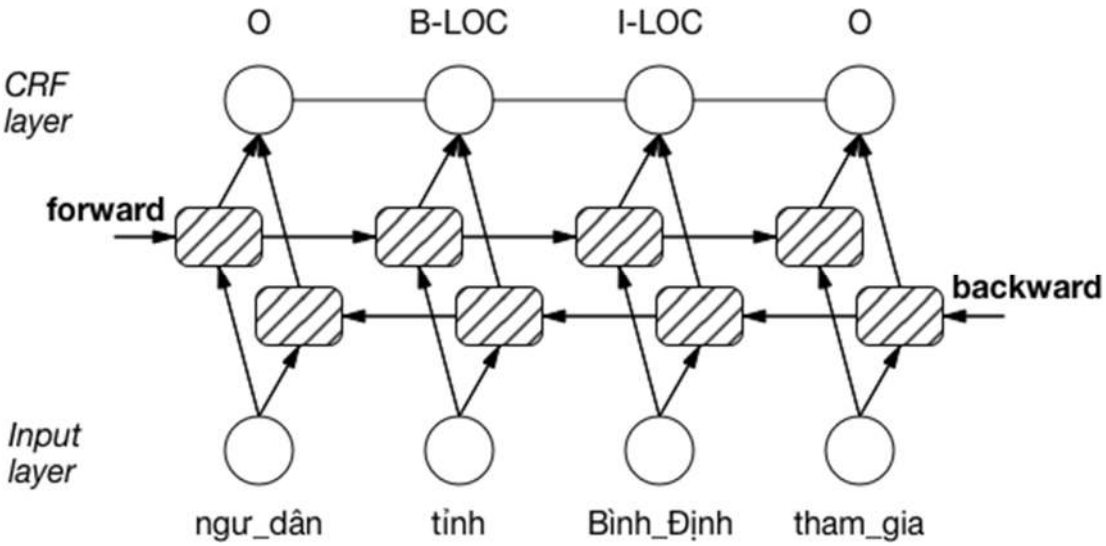
For English IDSF datasets, JointBERT-CRF holds the record for state-of-the-art results. For Vietnamese datasets, e.g., PhoATIS, JointIDSF achieves state of the art results, but the increase is trivial (<1%) despite the increase in training time. Thus, we choose JointBERT-CRF as our core framework, but with XLM-RoBERTa-base (XLM-RoBERTa) as the underlying language model.
The data is provided at a syllable level and thus might not be suitable for models that work on word-segmented data such as PhoBERT. Also, we notice a small portion of foreign terms in the data ('device', 'floor') and speculate that a cross-lingual model might learn better representations. Besides, we did experiments with PhoBERT and found XLM-RoBERTa to perform slightly better, so we chose XLM-RoBERTa as the base model.

Sentence Transformer

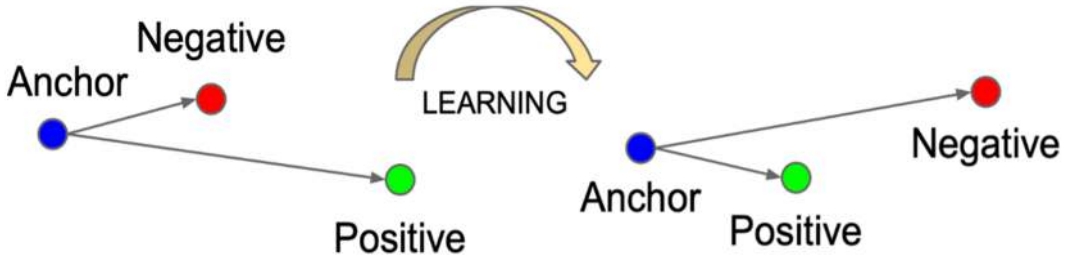
This framework provides an easy method to compute dense vector representations for sentences, paragraphs, and images. The models are based on transformer networks like BERT / RoBERTa / XLM-RoBERTa etc. and achieve state-of-the-art performance in the various tasks. Text is embedded in vector space such that similar text is close and can efficiently be found using cosine similarity.

Conditional Random Field (CRF)

Conditional Random Field (CRF) is a probabilistic model for structured prediction problems and has been applied successfully in many fields such as computer vision, natural language processing, and bioinformatics. In the CRF model, the nodes containing the input data and the nodes containing the output data are directly connected to each other, opposite to the architecture of LSTM or BiLSTM and especially in this problem BERT where the inputs and outputs are connected indirectly through memory cells.



Supervised Contrastive Learning



The main idea of Contrastive learning is to find pairs of similar and contrasting data features in a dataset. From there, with pairs of similar data, we can "pull" them closer to learn higher-level features of each other, and conversely contrasting pairs of data will be "pushed" away. . To do this, we will need to use similarity metrics to calculate the distance between the embedding vectors representing the data points. For example, if we already have 1 original data point called an anchor, then we can use different augmentation techniques to get 1 more variation from the original anchor called a positive sample, and the rest of the batch/dataset will are considered negative samples. Then the model will be trained to be able to distinguish a positive sample from a negative sample from a data cluster.

In the paper "Supervised Contrastive Learning" presented at NeurIPS 2020, researchers proposed a loss function called SupCon - Supervised Contrastive Loss, which aims to bridge the invisible gap between self-supervised learning and fully-supervised learning and allows contrastive learning to be applied to supervised learning problems. Still keeping the idea of contrastive learning, the SupCon function will try to take advantage

of the labeled data to pull together normalized vector embeddings of the same class and vice versa for embedding vectors of different classes. Having more labeled data simplifies the positive sample selection process and avoids false negatives.

$$\mathcal{L}_{\text{supcon}} = -\sum_{i=1}^{2n} \frac{1}{2|N_i| - 1} \sum_{j \in N(y_i), j \neq i} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{\sum_{k \in I, k \neq i} \exp(\mathbf{z}_i \cdot \mathbf{z}_k / \tau)}$$

3. Approach

3.1 Data exploration

The dataset consists of the train, dev, public test, and private test data with 1790, 392, 346, and 360 utterances accordingly. Each utterance (train, dev) has been annotated in slots (syllable-level) and intents.
An example of an utterance and its labels is as follow:

Utterance: Thay đổi đèn phòng ngủ sang màu xanh
Intent: set.color
Slots: B-command, I-command, B-device, I-device, I-device, O, B-color, I-color

Distributions of labels on the training and dev set, as well as distributions of sequence length between data portions, are quite similar.

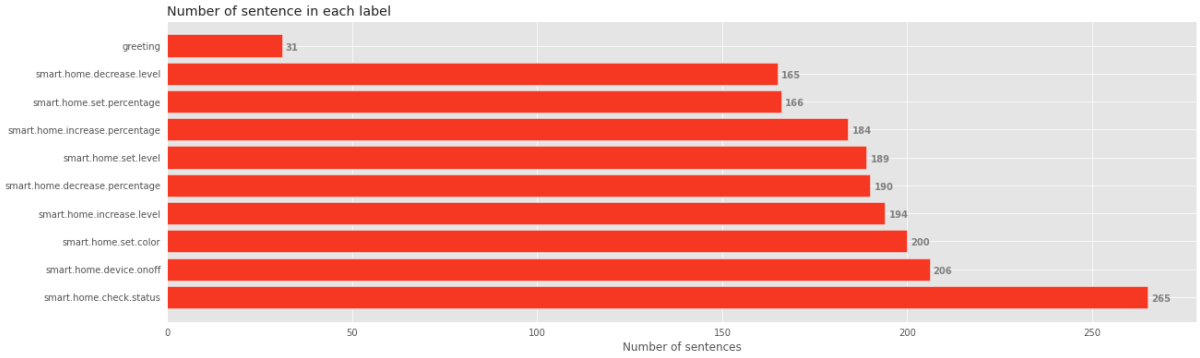


Figure: Train dataset intent

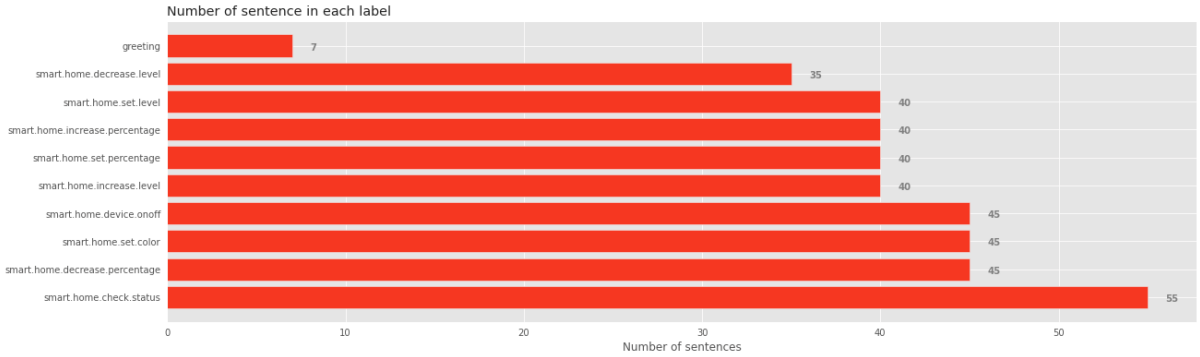


Figure: Dev dataset intent

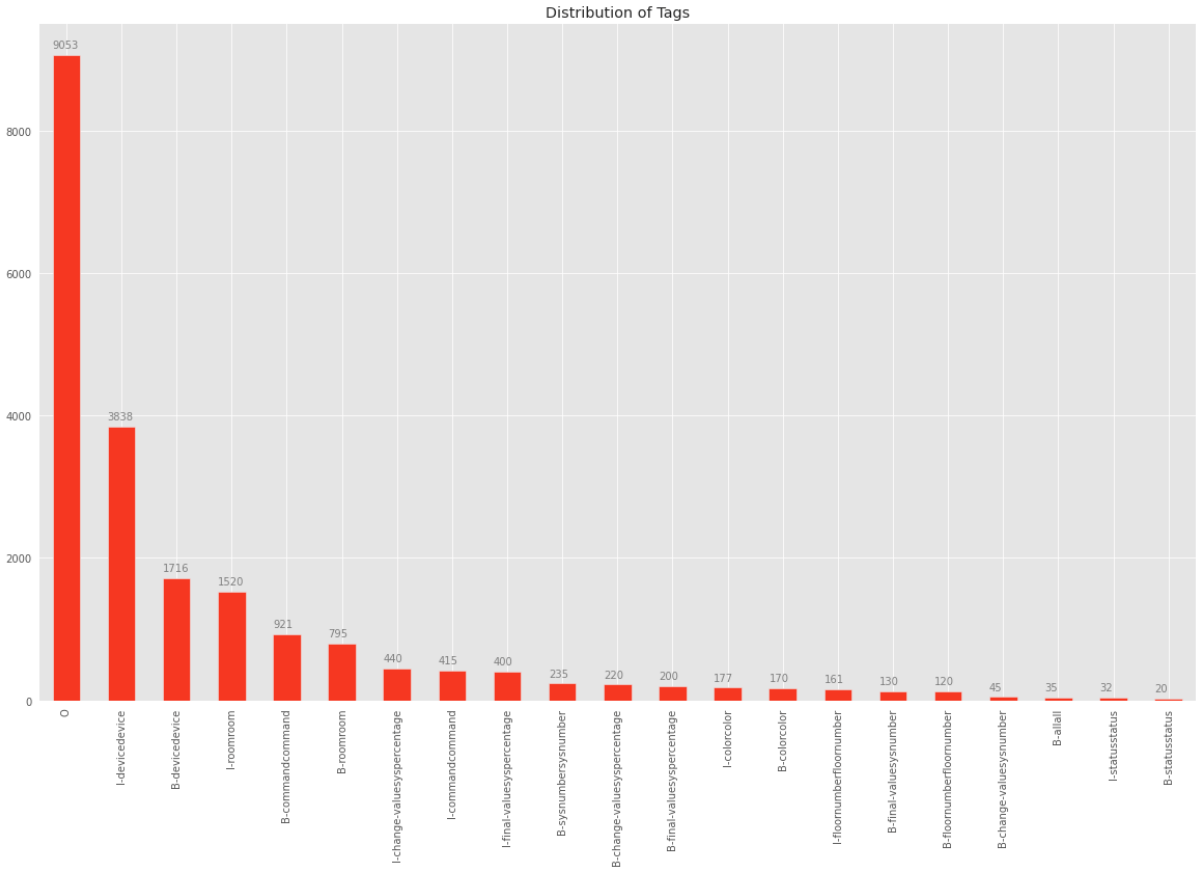


Figure: Train dataset slots include O-tokens

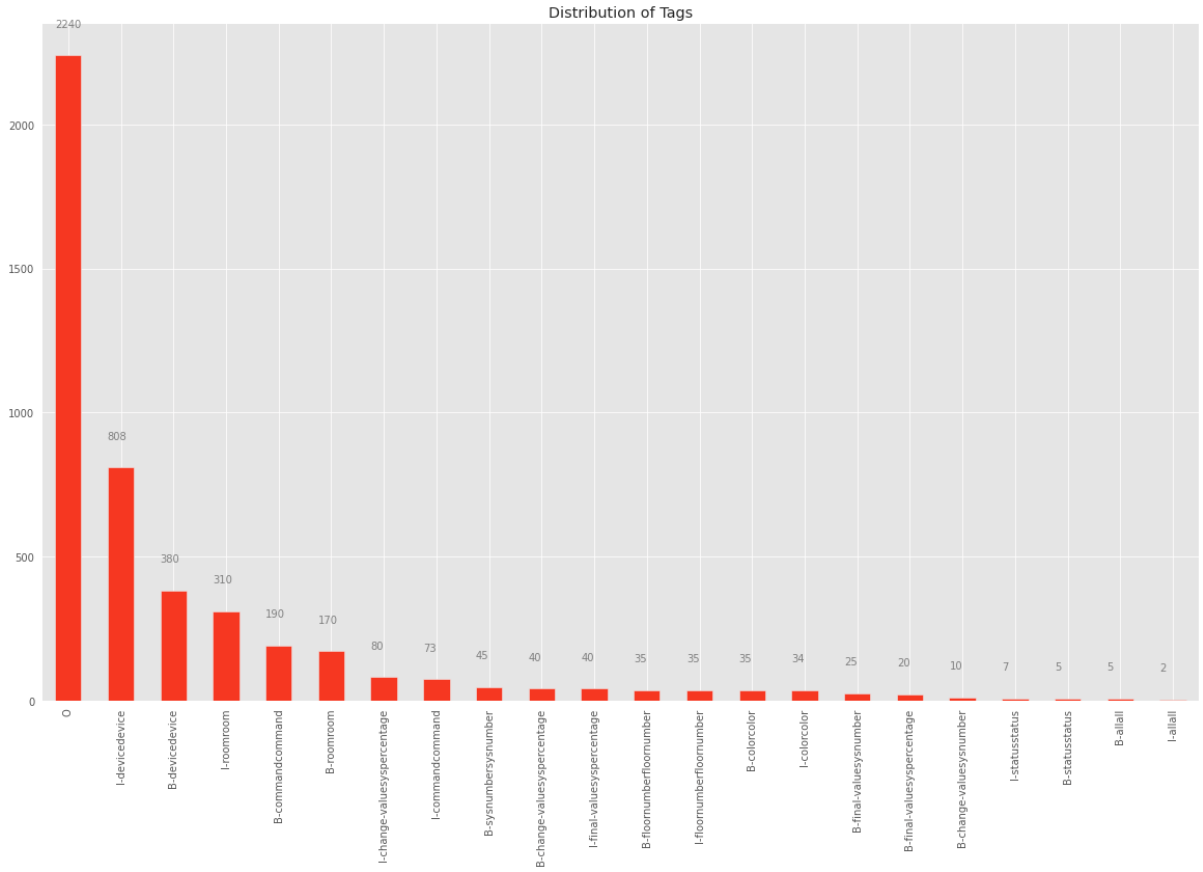


Figure: Dev dataset slots include O-tokens

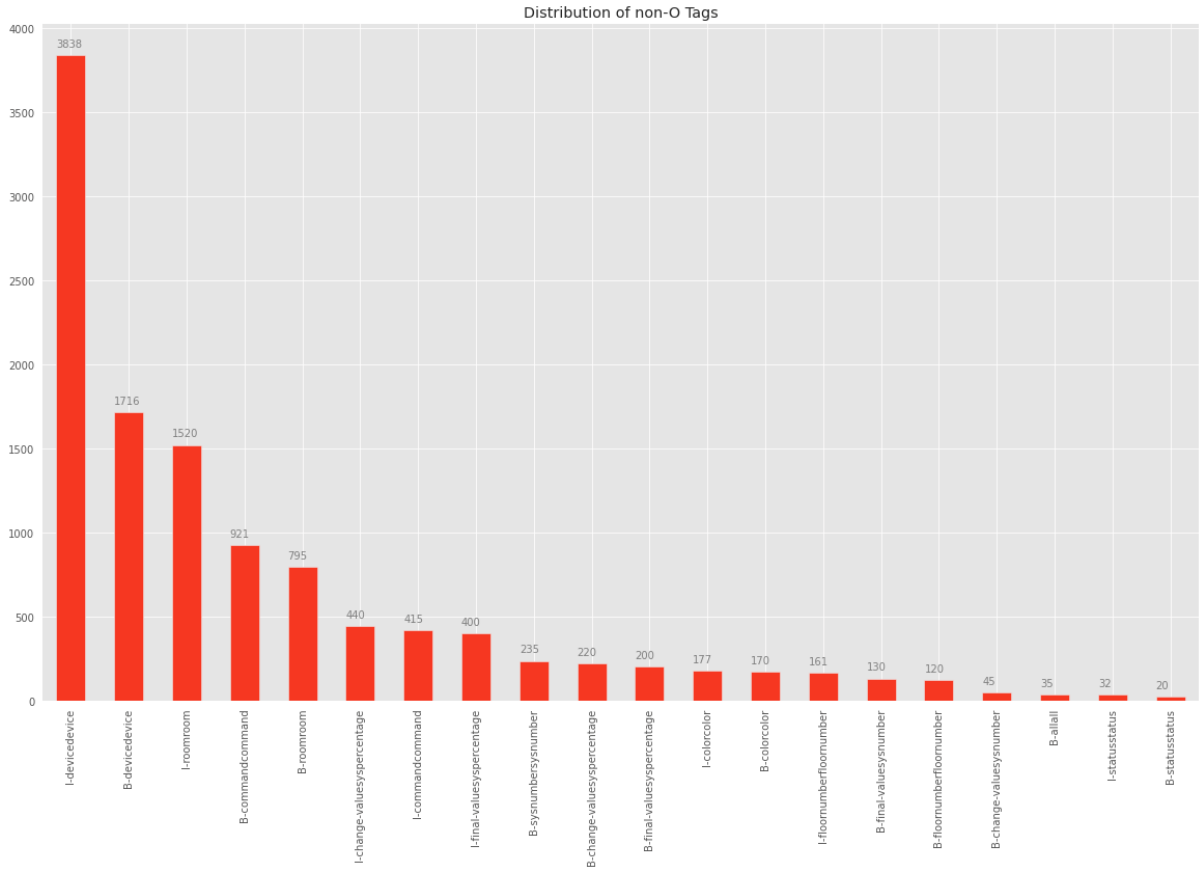


Figure: Train dataset slots non-O-tokens

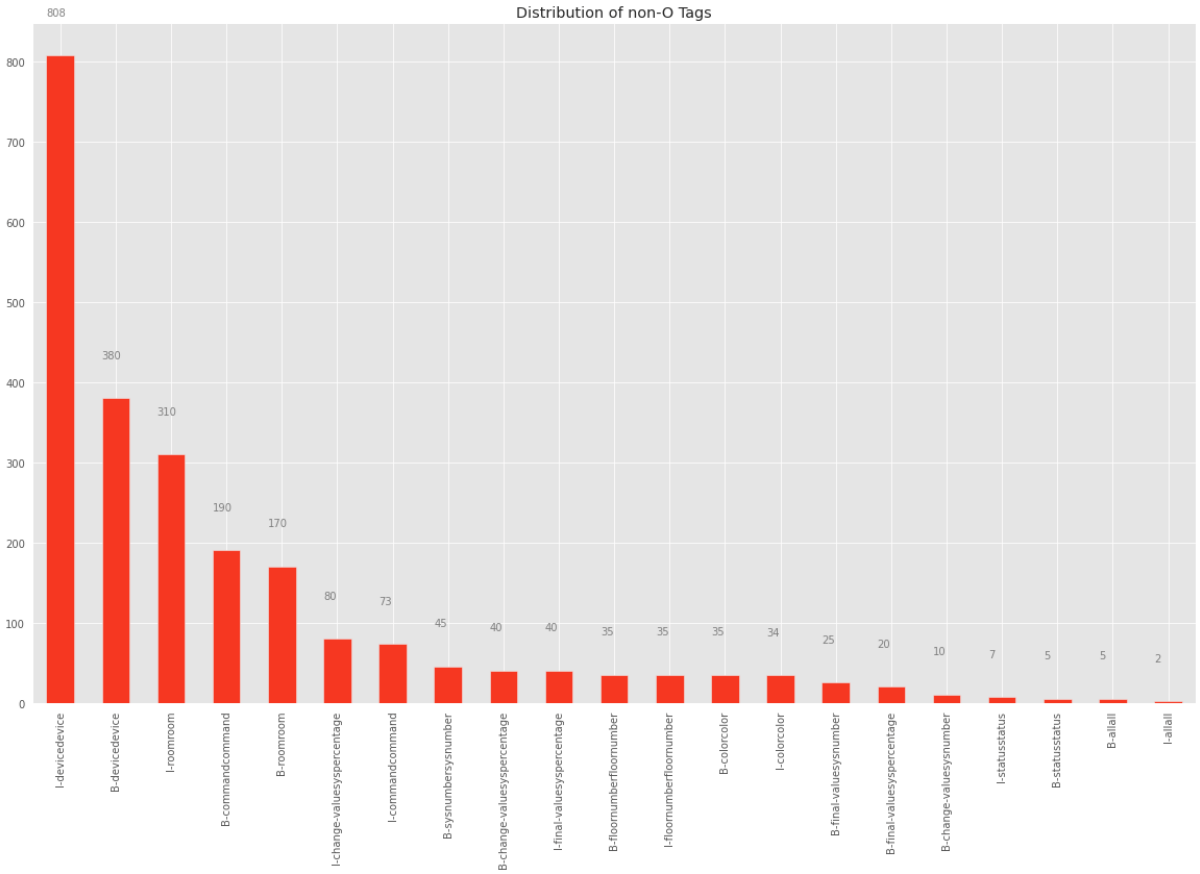


Figure: Dev dataset slots non-O-tokens

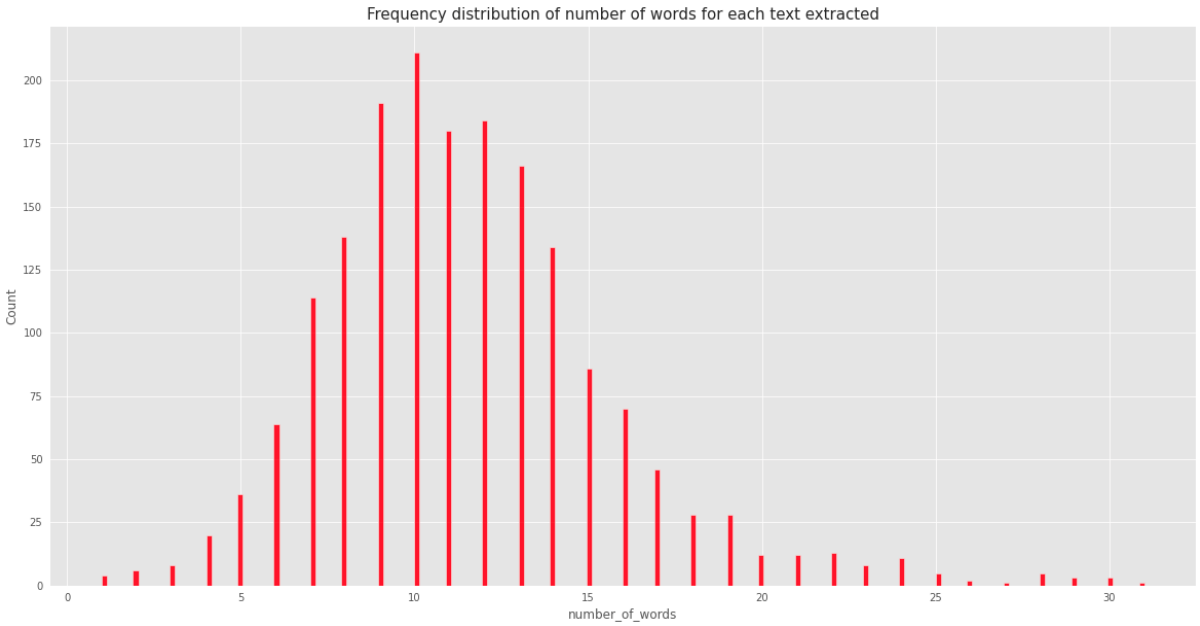


Figure: Train dataset sequence length

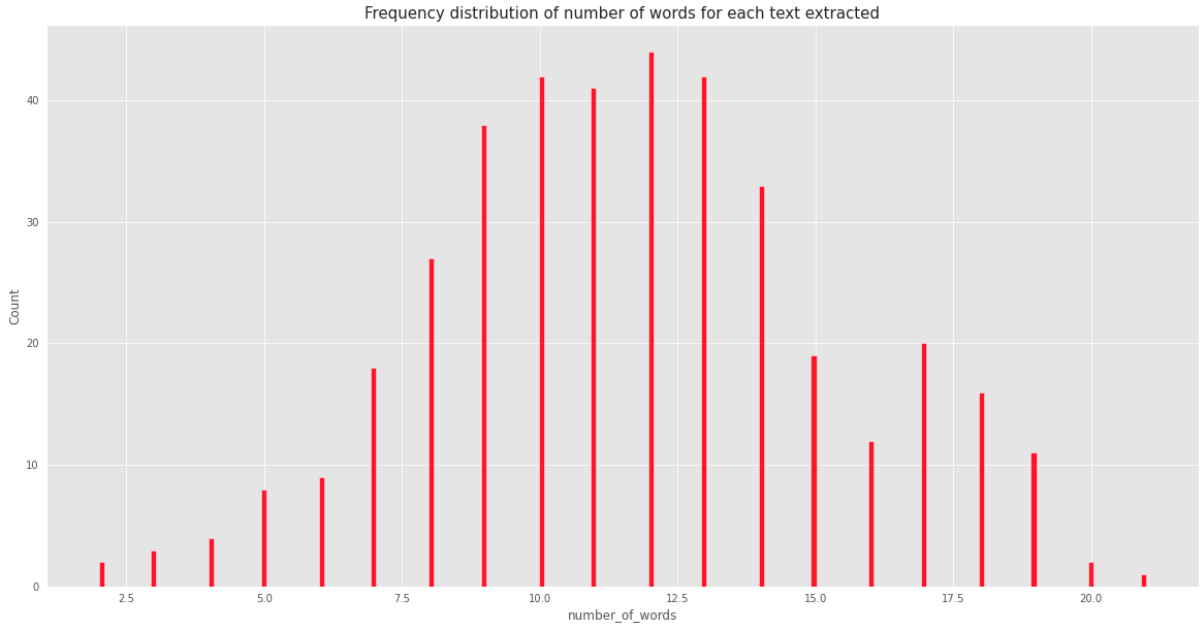


Figure: Dev dataset sequence length

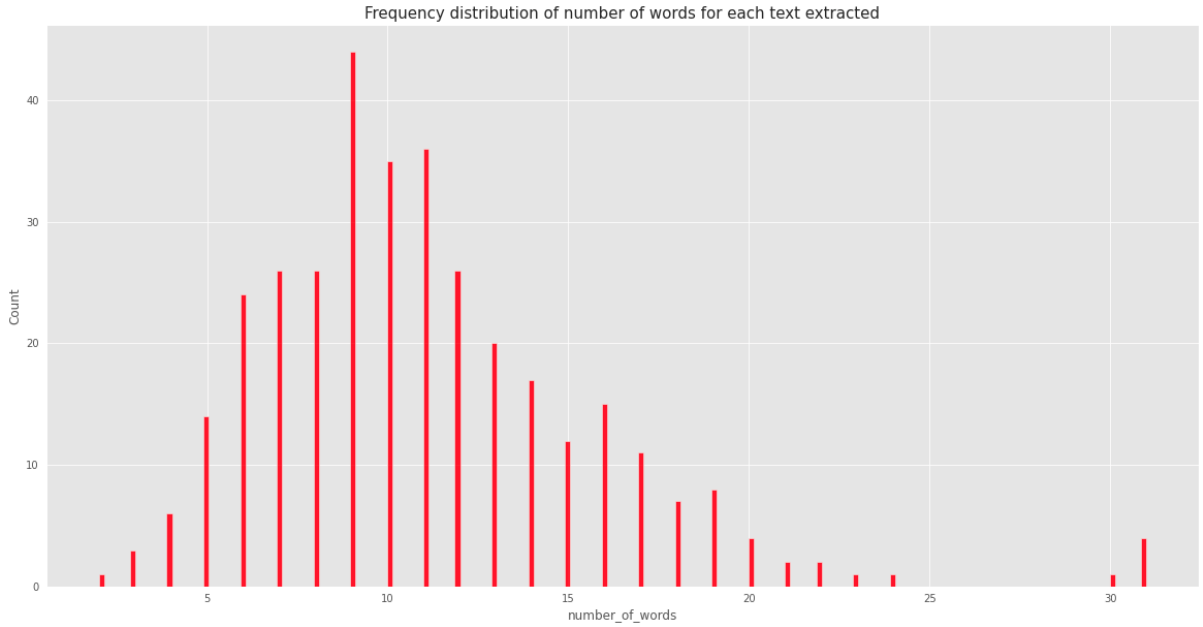


Figure: Public test dataset sequence length

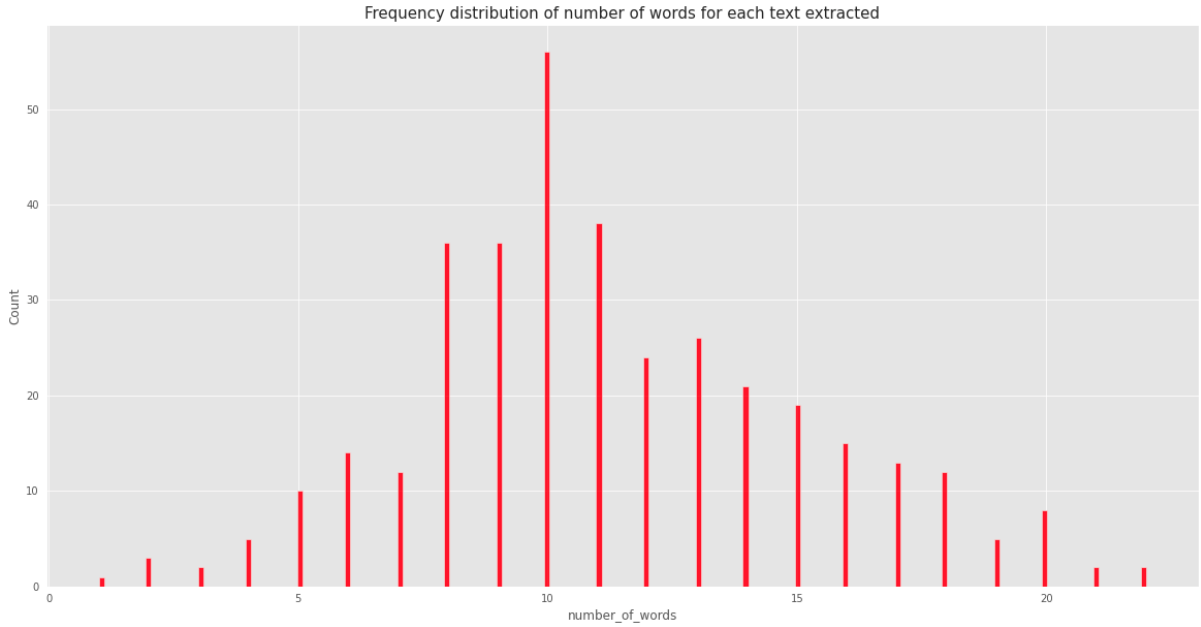


Figure: Private test dataset sequence length

There are no punctuations. No utterance exceeds 32 words.

3.2 Pre-processing

Label Fixing

We randomly sample some of the training instances and find that there are a lot of mislabeled samples (intent).
E.g.

Utterance	Original Intent	Corrected Intent
Tôi muốn bật đèn ngủ	set.percentage	device.onoff
Đóng thiết bị hộ anh	set.percentage	device.onoff
Giảm giúp mình điện thứ 3 nhé	decrease.percentage	decrease.level
Chào bạn ngày mới vui vẻ	set.percentage	greeting

To address the mislabeled samples, firstly, we collect words that appear more frequently in a certain intent, and less often in other intents. We call these words to pivot features. These features were collected through statistical analysis of words' frequencies in data.

E.g.

Intent	Pivot Features
decrease.level	giảm, mức, xuống,...
set.percentage	phần trăm, đặt,...
set.color	màu, sang, đổi,...

Next, we build rules to detect and reassign labels to noisy samples, based on the conflict between the appearance of these pivot features and incompatible intents.
For example, if the word ‘tăng’ appears in an utterance labeled ‘decrease’, that utterance is likely to be mislabeled (the right label might be ‘increase’) or we will remove utterances having ‘percentage’ intent without slots about ‘percentage’.
To mitigate the scenario that the rules might turn correct samples into misclassified ones, we further re-examine assigned utterances and update the rules accordingly.
Overall, 190 utterances’ intents were fixed based on the built-up rules (train & dev data).
For slot labeling, when performing word segmentation and aligning slots accordingly, we found some mislabelled nouns. For example, in one utterance, the word ‘bóng đèn’ was labeled [O, B-device], which wasn’t right, or we will relabel utterances with the label ‘final-valuepercentage’ mentioning ‘decreasing’ or ‘increasing’ to ‘change-valuepercentage’. As these cases were rare, we use simple rules to reassign such labels. A total of 10 utterances’ slots were fixed in this way.

3.3 Data augmentation

We experiment with three kinds of augmentations:

Random replacement (O-token)

- E.g. Bật **cho** tôi với => Bật **giúp** tôi với

Random add (between 2 O-tokens)

- E.g. Bật cho tôi => Bật cho **anh** tôi

Mixed up: Replace slot with word with similar slot types

- E.g. Bật **bóng đèn** lên => Bật **điều hòa** lên

For methods 1 and 2, we use a language model to fill in the blanks. The specific way is as follows:

Random replacement: We randomly select a location whose tag is O, then replace the position's word with <mask>. Finally, we use a language model (namely phoBERT) to fill in the <mask> position to get a new sentence.

- E.g. Bật **cho** tôi với => Bật **<mask>** tôi với => Bật **giúp** tôi với

Random add: We randomly select a position between two words with the tag O, then mark that spot as <mask>. Finally, we use a language model (namely phoBERT) to fill in the <mask> position to obtain a new sentence.

- E.g. Bật cho tôi => Bật cho **<mask>** tôi => Bật cho **anh** tôi

3.4 Model architecture

JointBERT-CRF

There are two main approaches: separate train models to predict the two tasks or use a joint model to perform joint training. JointBERT-CRF is a joint model.

The JointBERT-CRF model with intent-slot class attention is used to incorporate intent-context information into slot-filling through the intent label embedding layer.

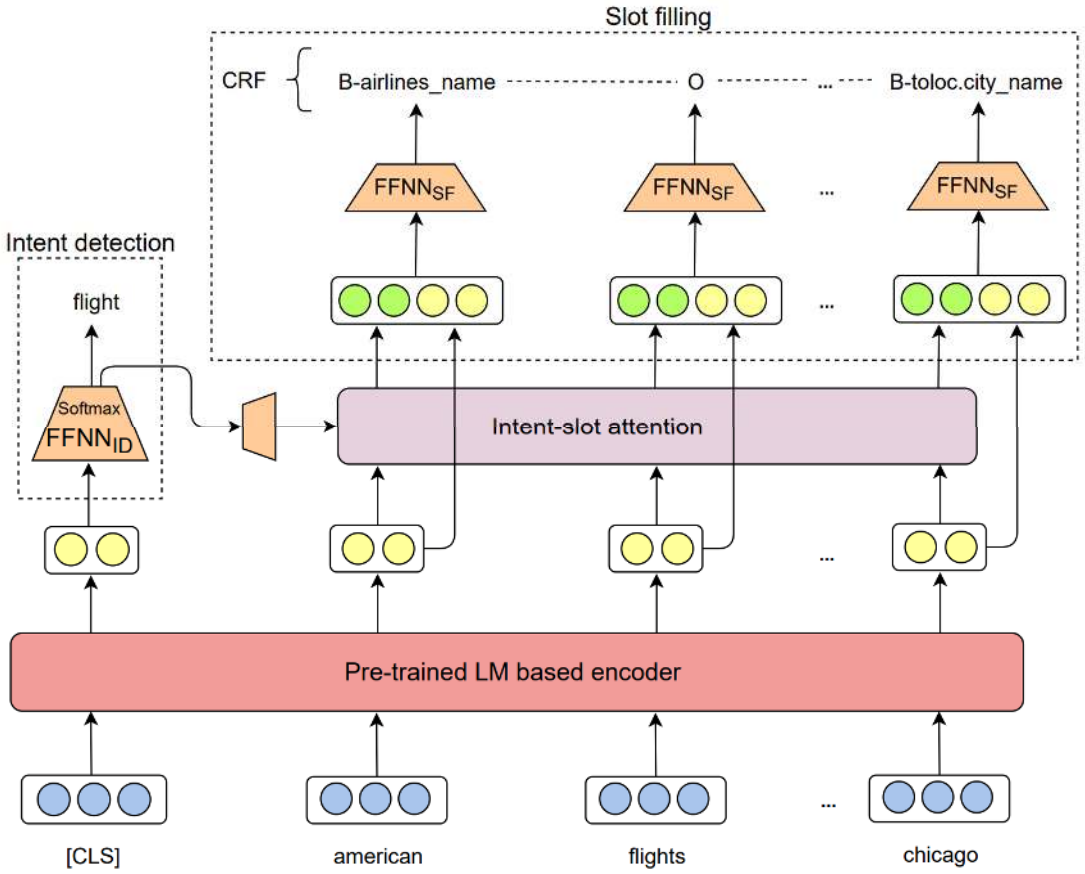


Figure: JoinBERT-CRF

The model architecture of BERT is a multi-layer bidirectional Transformer encoder based on the original Transformer model. A special classification embedding ([CLS]) is inserted as the first token and a special token ([SEP]) is added as the final token.

BERT can be easily extended to a joint intent classification and slot filling model. Based on the hidden state of the first special token ([CLS]), denoted *h1*, the intent is predicted as

$$y^i = \text{softmax}(\mathbf{W}^i \boldsymbol{h}_1 + \boldsymbol{b}^i)$$

For slot filling, we feed the final hidden states of other tokens *h2, . . . , hT* into a softmax layer to classify over the slot filling labels.

$$y_n^s = \text{softmax}(\mathbf{W}^s \boldsymbol{h}_n + \boldsymbol{b}^s), n \in 1 \ldots N$$

To jointly model intent classification and slot filling, the objective is formulated as

$$p(y^i, y^s|\boldsymbol{x}) = p(y^i|\boldsymbol{x}) \prod_{n=1}^{N} p(y_n^s|\boldsymbol{x})$$

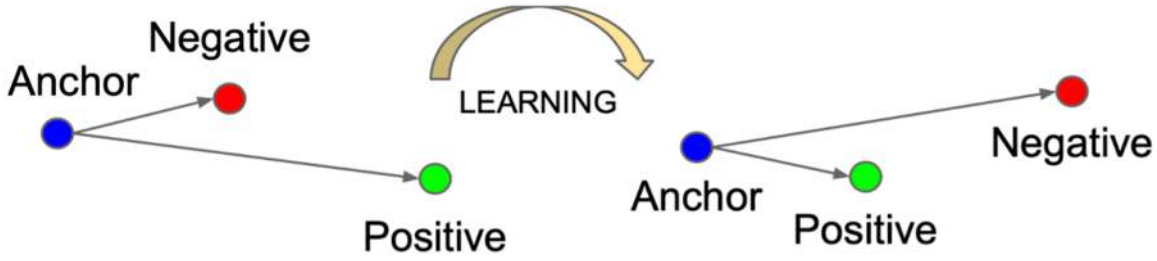
Slot label predictions are dependent on predictions for surrounding words. It has been shown that structured prediction models can improve slot filling performance, such as conditional random fields (CRF). Here we investigate the efficacy of adding CRF for modeling slot label dependencies, on top of the JointBERT model.

Contrastive learning (CL) for improving Intent Classification

Issue: Some intents seem to share something in their meanings

E.g. **smart.home.decrease.percentage** and **smart.home.decrease.level.**

This causes unseparated representations in embedding space. Therefore, when classifying these intentions together, confusion will easily occur. Therefore, we propose Contrastive learning method for better representations.



We are provided with a description of the intents by the organizers.

E.g. smart.home.set.percentage: "Bật thiết bị ở mức phần trăm nhất định".

We use Sentence BERT to turn those descriptions into representation vectors ($repi$). Intuitively, we want the representation of the sample of intent i to be closed to $repi$ and far away from $repj$ ($j \neq i$).

Formally, the objective function for Contrastive Learning loss for 1 sample s:

$$\mathcal{L}_{CL} = -\log \frac{\exp\left(sim(s', rep_i)\right)}{\sum_{j \neq i} \exp\left(sim(s', rep_j)\right)}$$

where $s' = <CLS> representation\ vector\ of\ s$

4. Training and Experiments

Metrics

Sentence accuracy: A sentence is evaluated as correct when the intent is predicted correctly and all slots in the sentence are also correctly predicted. This is a rather difficult assessment, requiring exactly all positions in the sentence.

Augmentation

After performing augmentation methods on the train+dev set, we obtain a data set as follows:

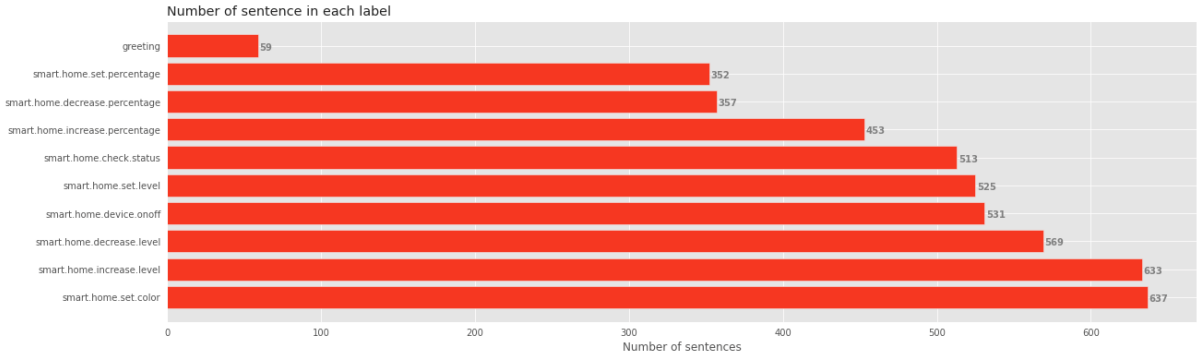


Figure: Train + dev dataset intent

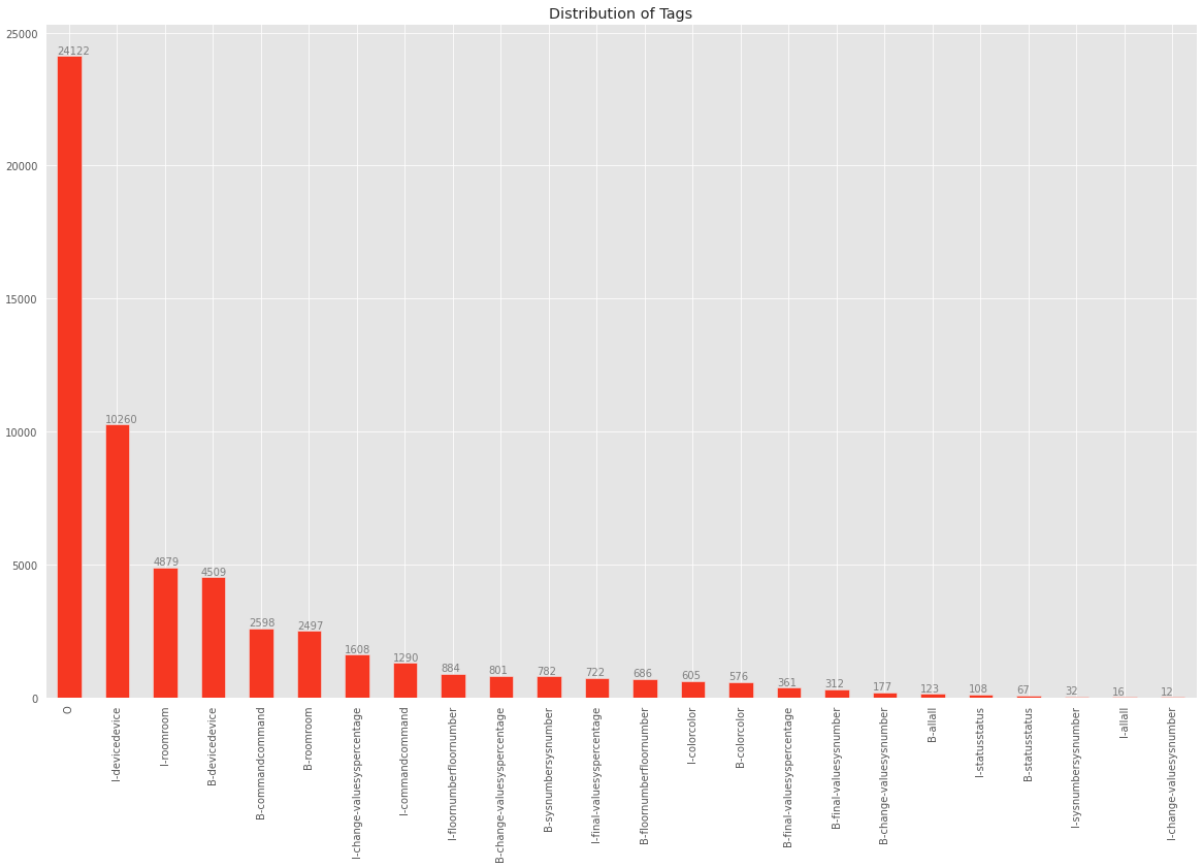


Figure: Train + dev dataset slots include O-tokens

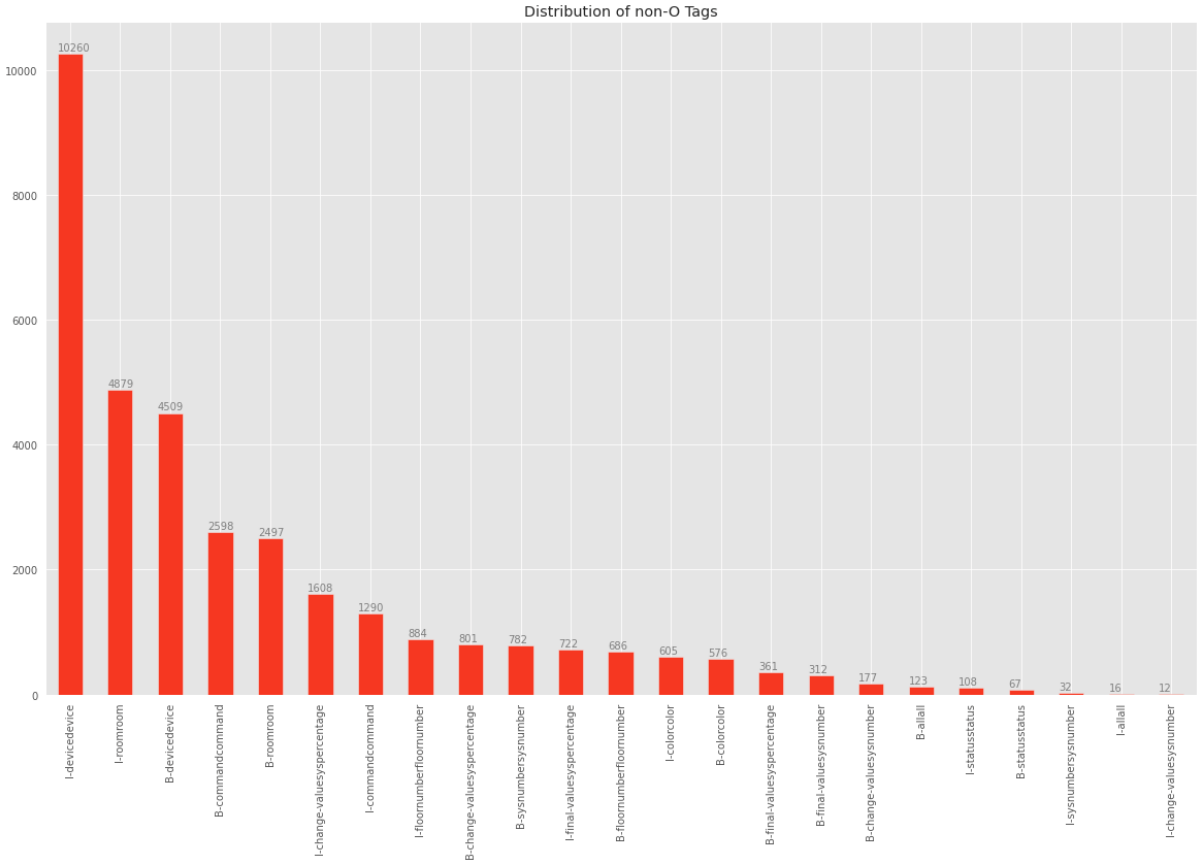


Figure: Train + dev dataset slots non-O-tokens

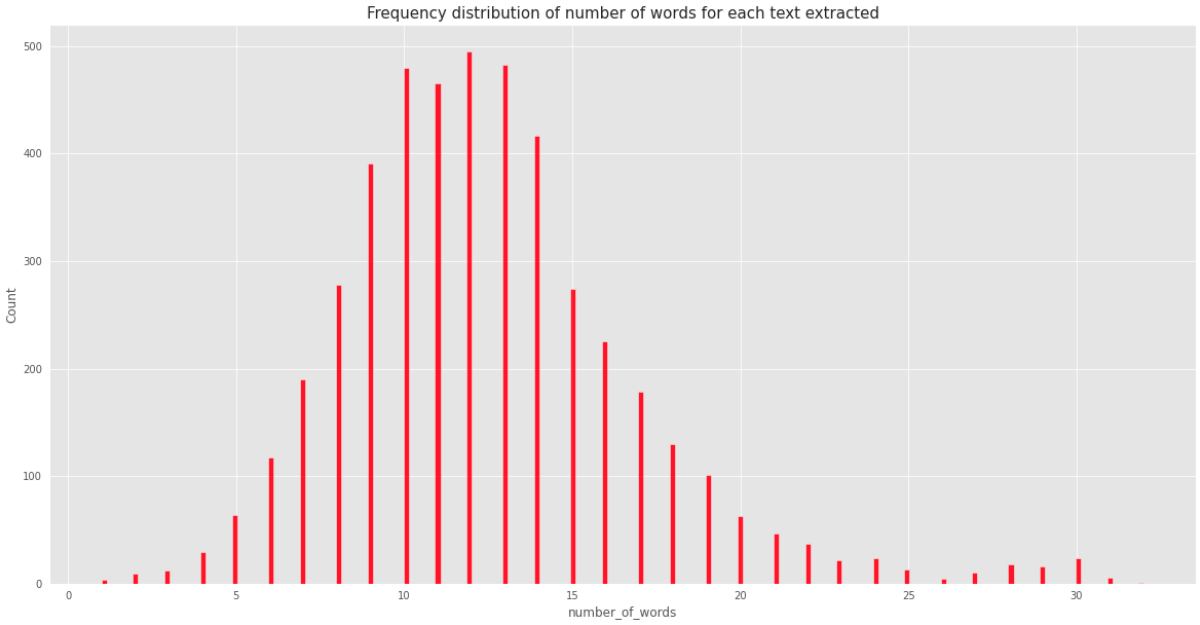


Figure: Train + dev dataset sentence length

Training parameters

For final submissions on the public test and private test, we used both the training and dev data (reassigned labels) for training.

Optimizer	AdamW
Num epochs	30
Batch size	32
Intent loss coefficient	0.5
Dropout rate	0.2
Learning rate	3e-5
Tunning metric	Mean Intent-Slot Accuracy

Public test

Model	Intent F1	Slot F1	Mean Intent-slot	Sentence accuracy
SVM and BiLSTM-CRF	80,52	90,43	85,48	59,44
JoinBERT-CRF	84,95	92,64	88,79	70,42
JoinBERT-CRF + Augmentation	86,42	94,70	90,56	72,54
JointBERT-CRF + Contrastive Learning + Augmentation	**92,89**	**95,01**	**93,95**	**79,48**

Private test

Model	Intent F1	Slot F1	Mean Intent-slot	Sentence accuracy
SVM and BiLSTM-CRF	76,36	78,52	77,44	48,16
JoinBERT-CRF	80,32	88, 23	84,27	65,00
JoinBERT-CRF + Augmentation	86,94	92,42	89,69	67,78
JointBERT-CRF + Contrastive Learning + Augmentation	**87,61**	**93,26**	**90,43**	**70,83**

5. Summary

Finally, after augmenting, the data has become more diverse, so the Slot-F1 Score has clearly improved. In addition, the Supervised Contrastive Learning method has made it easier to classify intents, easily confused intentions have been correctly classified. Because the dataset is quite challenging and the metric given is very difficult, our scores are still not as high as we would like. We will continue to research and come up with methods to improve this problem.

References

BERT for Joint Intent Classification and Slot Filling, Chen et al.

Unsupervised Cross-lingual Representation Learning at Scale, Alexis Conneau and Kartikay Khandelwal et al.

Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, Nils et al.

MINILM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers, Wang et al.

Supervised Contrastive Learning, Khosla et al.

SentenceTransformers Documentation