Address Elements Extraction

Shopee Code League 2021
Data Science Round

Problem Statement

- Extract **Point of Interest (POI)** and **Street** from Indonesian address text
- Enable accurate geocoding for delivery optimisation, improving Shopee customer experience
- Most addresses are **unstructured, incomplete,** and can even be **misspelled**
- 300,000 training set + 50,000 test set

Raw address	POI	Street
cipinang besar selatan lintas ibadah, cipi jaya 1a no 3 rw 7 13410 jatinegara	lintas ibadah	cipinang jaya 1a
puri kemb timur	<None>	puri kembang timur

Named Entity Recognition (NER) Training Pipeline

1. Split sentences into **individual words**
2. **Align** the POI and Street with the input words
3. Label each word with a **tag** for prediction
4. Fine-tune a pretrained language model on a **multi-class** **single-label** prediction on each word
5. Use the predicted labels to construct the POI and Street names, fixing **incomplete** or **misspelled** words if necessary

Split Sentence + Aligning POI/Street

- Simple text cleaning and word splitting using RegEx
- Simple linear substring matching algorithm to find the POI/Street positions inside the raw address:
    - Use **prefix match** (with minimum prefix length) for each word instead of exact match to account for **incomplete** or **misspelled** words
    - Sometimes fail to align if there are **no matches** (i.e misspelled at prefix) or **too many matches** (i.e POI/Street names too short), but only **~1000 rows**

Raw address	**law stat**, **hayam wuruk**, sumerta kelod denpasar timur
POI	lawson station
Street	hayam wuruk

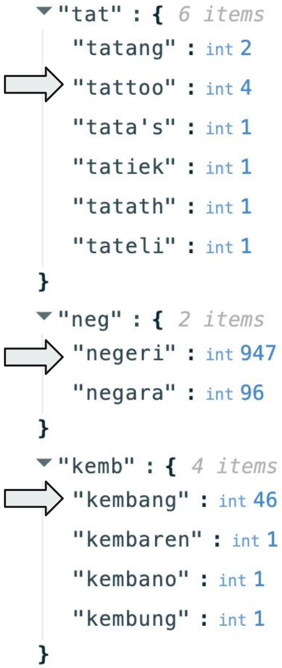
IOBES + {SHORT} Tagging Scheme

- I: inside
- O: outside
- B: beginning
- E: ending
- S: single
- SHORT: need fixing

Raw address	**law stat**, **hayam wuruk**, sumerta kelod denpasar timur
POI	lawson station
Street	hayam wuruk
Individual words	['law', 'stat,', 'hayam', 'wuruk,', 'sumerta', 'kelod', 'denpasar', 'timur']
Individual tags	['B-POI-SHORT', 'E-POI-SHORT', 'B-STR', 'E-STR', 'O', 'O', 'O', 'O']

SHORT Word Reconstruction Dictionary

- Record all words with tags **SHORT** into a one-to-one dictionary
- Use this dictionary to fix words with tags **SHORT** during validation and testing
- Not an “ideal” approach
    - Some **SHORT** words might not appear in the dictionary
    - Some **SHORT** words might have multiple mappings, so pick one with the **highest frequency**
- But simple and surprisingly effective enough!



```
▼"tat" : { 6 items
    "tatang" : int 2
⇨   "tattoo" : int 4
    "tata's" : int 1
    "tatiek" : int 1
    "tatath" : int 1
    "tateli" : int 1
}
▼"neg" : { 2 items
⇨   "negeri" : int 947
    "negara" : int 96
}
▼"kemb" : { 4 items
⇨   "kembang" : int 46
    "kembaren" : int 1
    "kembano" : int 1
    "kembung" : int 1
}
```

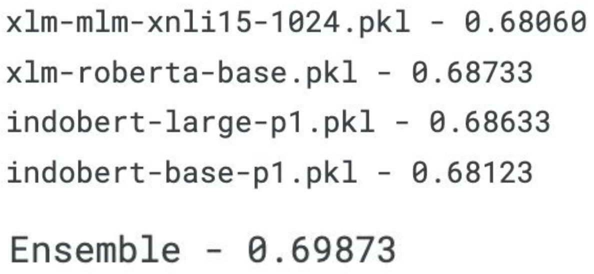
Data Augmentation

- **Swap POI/Street phrases** within a sentence, or across sentences randomly
- Increase training data size ~**2x**

Model Training

- Fine-tuning suitable pretrained transformer language models on downstream task:
  - IndoBERT (Indonesian)
  - XLM (multilingual)
- Standard pipeline:
  - Adam optimizer
  - Cross Entropy Loss
  - 5 epochs is enough for model convergence
- Techniques for stable and efficient training:
  - Cyclic learning rate scheduler with warmup
  - Mixed Precision Training

- Average logits from multiple models for ensembling, **~0.02** accuracy increase

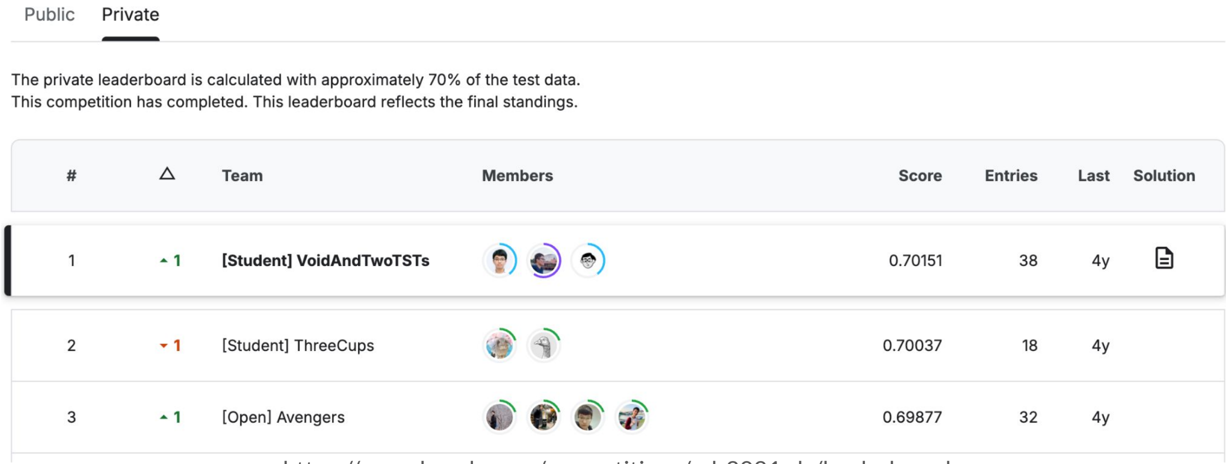


```
xlm-mlm-xnli15-1024.pkl - 0.68060
xlm-roberta-base.pkl - 0.68733
indobert-large-p1.pkl - 0.68633
indobert-base-p1.pkl - 0.68123

Ensemble - 0.69873
```

Result

- Final accuracy: ~70%
- Ranked **1st** out of 1034 teams



The private leaderboard is calculated with approximately 70% of the test data.
This competition has completed. This leaderboard reflects the final standings.

#	△	Team	Members	Score	Entries	Last	Solution
1	▲ 1	**[Student] VoidAndTwoTSTs**		0.70151	38	4y	📄
2	▼ 1	[Student] ThreeCups		0.70037	18	4y	
3	▲ 1	[Open] Avengers		0.69877	32	4y	

https://www.kaggle.com/competitions/scl-2021-ds/leaderboard

Reflections

- Experimenting with **data processing and augmentation** led to the highest accuracy improvement
- This was done in 2021, so there should be much better state-of-the-art pretrained models now
- Maybe can even consider fine-tuning a Large Language Model (LLM) and model the problem as a text generation task instead of classification task

Potential point for improvements:

- More data augmentation can be considered such as using synonyms, or restructuring the sentences
- The fixing pipeline can use another language model that takes into account the the address before fixing