Data Mining Lab2

112061536 施睿為

1 Pre-processing

1.1 Load data.

Load data from provided csv and Json files as dataframes and merge them.

	tweet_id	text	_score	_index	hashtags	_crawldate	_type	identification	emotion
0	0x376b20	People who post "add me on #Snapchat" must be ...	391	hashtag_tweets	[Snapchat]	2015-05-23 11:42:47	tweets	train	anticipation
1	0x2d5350	@brianklaas As we see, Trump is dangerous to #...	433	hashtag_tweets	[freepress, TrumpLegacy, CNN]	2016-01-28 04:52:09	tweets	train	sadness
2	0x28b412	Confident of your obedience, I write to you, k...	232	hashtag_tweets	[bibleverse]	2017-12-25 04:39:20	tweets	test	NaN
3	0x1cd5b0	Now ISSA is stalking Tasha 😂😂😂 <LH>	376	hashtag_tweets	[]	2016-01-24 23:53:05	tweets	train	fear
4	0x2de201	"Trust is not the same as faith. A friend is s...	989	hashtag_tweets	[]	2016-01-08 17:18:59	tweets	test	NaN

1.2 Check duplicated data and missing values.

There’s no duplicated data and missing values.

1.3 Remove meaningless attributes.

Such as 'identification', '_crawldate', '_index', and '_type' in training dataset.

```
df_train = df_train.drop(columns=['identification','_crawldate','_index','_type'])
df_test = df_test.drop(columns=['emotion','identification','_crawldate','_index','_type'])
```

For my model, I didn’t utilize 'hashtags' attribute.

1.4 Store data in ‘dataset’ module.

It helps improve code quality, maintainability, scalability, and performance.

```
train_dataset = Dataset.from_pandas(df_train_no_hashtags)
dataset = DatasetDict({
    'train': train_dataset
})
```

2 Feature Engineering

2.1 Tokenizer.

Our training data is English sentences which is str type. The model I used is ‘distilbert-base-uncased’.

1. **Uncased**: The ‘uncased’ aspect means that the tokenizer does not differentiate between uppercase and lowercase letters. For example, "Dog" and "dog" are

treated as the same word. This helps to reduce the vocabulary size and is useful for certain NLP tasks where the case of letters is not significant.

2. **WordPiece Tokenization**: This tokenizer uses the WordPiece tokenization method. It first splits the text into words and then further splits these words into smaller units, known as tokens. This is beneficial for handling unknown or rare words, as they can be broken down into smaller, known subunits.
3. **Special Tokens**: Like other BERT-like models, DistilBERT's tokenizer uses special tokens, such as:
   - **[CLS]**: Added at the start of each input sequence. In classification tasks, the hidden state of this token is often used to represent the overall meaning of the entire input sequence.
   - **[SEP]**: Used to separate different sentences or paragraphs.
   - **[PAD]**: Used for padding sequences that are shorter than the maximum length.
   - **[UNK]**: Represents an unknown token.
4. **Maximum Sequence Length**: The tokenizer has a limit on the maximum sequence length. For 'distilbert-base-uncased', it is typically 512 tokens. Inputs longer than this length are truncated.
5. **Output**: The output of the tokenizer is a sequence of token IDs. These IDs correspond to the index of each token in the DistilBERT vocabulary. These ID sequences are then used as input to the model.

2.2 One-hot encoder.

Because the labels are 'nominal' type data. They don't have the concept of order, spacing and proportion, so it better to encode them as one-hot encoding, which ensures the model won't use distance information.

```
X = [['anticipation'], ['sadness'], ['fear'], ['joy'], ['anger'], ['trust'],['disgust'], ['surprise']]
encoder.fit(X)
```

3 Model Explanation

The model I used is '**distilbert-base-uncased**'.

1. Distilled version of the original BERT:

   Use a technique called 'knowledge distillation (KD)' to get distilled version BERT. The main idea is that the original BERT is well-trained and powerful but consumes too much computing resource. To speedup the inference process, we would like a compact version model without performance drop. KD utilize the original model (teacher)'s logits to train a smaller model (student) so that the student model can learned better than training from scratch. Finally, we get a small but powerful BERT model. Its architecture is like BERT but with fewer layers. For instance, distilbert-base-uncased typically has 6 transformer layers, while the base BERT model has 12.

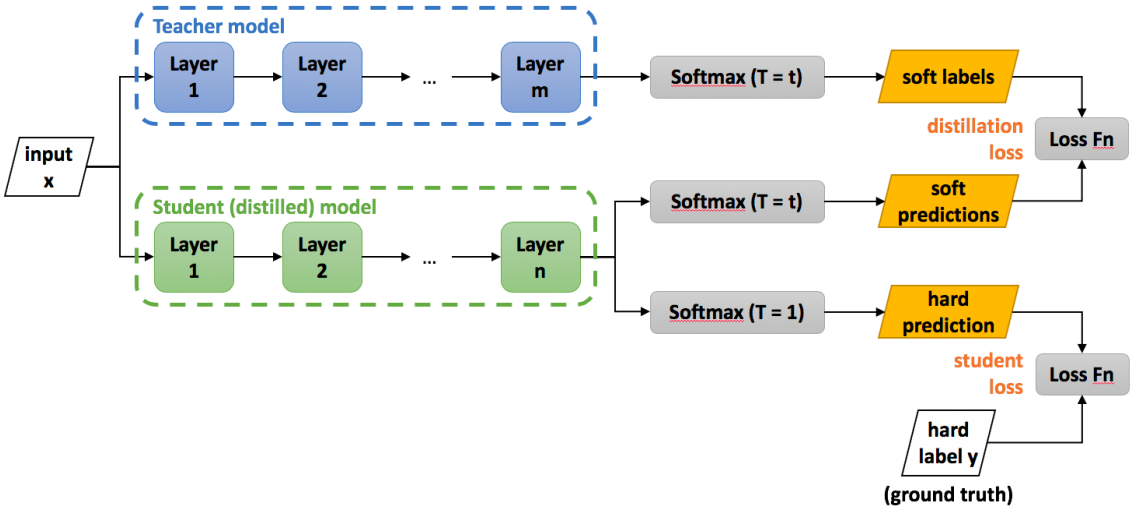

Fig 1. The pipeline of knowledge distillation.

4 Finding

4.1 More training epochs won't perform better.

I tried to train the model with the same hyper-parameters except the number of epochs. The results reveals that it better to fine-tune Bert models in 5-8 epochs, or the accuracy will drop significantly. For example, 3% drop for 1 epoch and 5% drop for 50 epochs.

4.2 Tags with '@' and '#' in tweets are informative.

I trained the same model by data without tags with '@' and '#' and hope that can reduce the noise in training data. However, they perform worse. The reason is that some of these tags contain critical information. For example, @peace and #happyWeekend. The better way is creating a mechanism to filter the meaningless tags, such as calculate the correlation with labels by Google's open source word2vector model.