Report

Prakapenka Tsimafei, March 2022.

Introduction

This project is aiming to implement honorifics (T/V) distinction for translating English to Russian. It is inspired by the article 'Controlling Politeness in Neural Machine Translation via Side Constraints'. The main idea is appending a special token to source sentences (<T>, <V>) based on target side information, and training neural-based translation model. Then, at inference we can control the politeness of an output translation by prepending a special token to source sentence.

Several main challenges were solved in this project:

1. **T/V distinction**. First, we should have an apparatus for classifying sentences as T, V, or neutral. I have developed two detectors: token-based and grammar-based.
2. **Data Curation**. In general, neutral sentences are dominating in all EngRu corpora, so I crafted special subcorpora with larger presence of T/V sentences.
3. **Translation**. I used JoeyNMT as a translation framework and followed the standard data preprocessing pipeline: tokenisation, truecasing, subword sampling (BPE).
4. **Evaluation**. I use several translation quality metrics: BLEU and more advanced BLEURT. Also, I report T/V count in reference and translated files to measure T/V control and not just general translation quality.

In the following sections, I will describe in detail T/V detection techniques, data collection, model training, and evaluation.

T/V detection

T/V categories are mainly distinguishable by specific pronouns (you), determiners (yours), and verb forms. I have created an abstract base class TVDetector and two actual implementations:

1) ***Token-based -*** `code/tv_labeling.py/SimpleTVDetector`

It simply seeks a substring of T/V pronouns and determiners in different forms. I have manually collected different forms of T/V analogues of ‘you’ and ‘your’ in Russian. Then, I split every input line by space and look if there is T or V token present. If both tokens are present, I mark this sentence as neutral.

2) ***Grammar-based -***

There are two steps in the grammar-based parsing approach.

First, I do morphosyntactic parsing for the Russian data using the DeepPavlov framework. I use Jupyter Notebook format (`code/notebooks/parse_file_to_conll.ipynb`) for implementation and run it using Google Colaboratory. This notebook is designed for the single target - parse test file in Russian to CoNLL format. The output of parsing is later used to label sentences with T/V tags based on a set of heuristics. Sample parsing output of Russian translation for the sentence `You 've never been caught .`:

```
# sent_id = 3
# text = Тебя не ловили .
1  Тебя   ты PRON   _  Case=Acc|Number=Sing|Person=2  3  obj    _  _
2  не не PART   _  _  3  advmod _  _
3  ловили ловить VERB   _  Aspect=Imp|Mood=Ind|Number=Plur|Tense=Past|VerbForm=Fin|Voice=A
ct  0  root   _  _
4  .  .  PUNCT  _  _  3  punct  _  _
```

Second, I parse CoNLL files and look for T/V specific Russian pronouns, but with this approach, I get rid of collecting various forms of the same word and refer only to part-of-speech and lemma.

Also, I look for verbs in form of the second person and plural or single numbers. Implementation is available here - `code/conll_tv_detector.py/ConllTVDetector`.

I have also run a comparison of approaches, and concluded that a simple token-based detector identifies ~79% of T/V labels compared to the grammar-based one (`code/helper.py/compare_tv_detectors`). Rest 21% is verb-based T/V detection. However, the grammar-based approach requires us to run heavy BERT-based parsing (~1h) for each produced test file. I have used SimpleTVDetector for training data markup and ConllTVDetector for all translation files.

Data

I use an English-Russian corpus of 1 million sentences provided by Yandex as train data. This corpus was collected automatically from various internet sources with data from 2011 to 2013. I used a simple token-based T/V detector to label sentences. There are 65970 V-sentences and 7948 T-sentences in 1m corpus, so I have decided to use all available T-sentences (~8k), the first 22k of V-sentences, and the first 100k of neutral sentences. Implementation is here: `code/helper.py/create_tv_corpus`.

The test dataset was crafted from manually annotated sources for solving the deixis problem for Voita et. al, 2019. The original data was taken from the OpenSubtitles dataset, so it is mostly informal dialogues. The most important feature of this data is Russian translations varying only in T/V for the same English sources. My processing code is here: `code/helper.py/reformat_deixis_files`

Model

I have used https://github.com/joeynmt/joeynmt as a base translation framework. I was massively relying on my Russian-Belarusian translator for this project. I use the Jupyter Notebook format for implementation and run it using Google Colaboratory. The main goals of this notebook are data preprocessing and model training with the GPU backend. We append <T> and <V> tokens at the beginning of the English sentence based on information provided by reference Russian translation. Both models (with and without T/V labelling) were trained for 30 epochs.

Interactive sentence and file translation using trained checkpoint is implemented with a separate Colab Notebook. For demo running please copy TV_model or base_model folder to your Google Drive, tweak runtime_path to reflect the actual location, and run the demo. Test file translation (~10k sentences) with Google Colaboratory and GPU runtime takes 5-7 minutes.

Evaluation

I use several translation quality metrics: BLEU and more advanced BLEURT. Also, I report T/V count in reference and translated files to measure T/V control and not just general translation quality.

Let us start with some example of T/V control:

English Source	***Russian Translation***	***Comments***
Where are you going ?	Где **идет**?	neutral translation, no T/V signals
<V> Where are you going ?	Где **вы** иде**те?**	polite translation, V signal is in pronoun and verb form
<T> Where are you going ?	Где **ты** иде**шь?**	informal translation, T signal is in pronoun and verb form

Next, I evaluate test files. The pipeline is the following:

1) Produce translation file using Demo Notebook

2) Do syntactic parsing using DeepPavlov Notebook and download the resulting .conll file

CoNLL parsing for a single test file runs for ~60 minutes on Google Colab.

3) Run BLEURT evaluation using BLEURT Notebook and download resulting .bleurt file

BLEURT evaluation for single test file runs for ~80 minutes on Google Colab.

4) Put all three files (translation, .conll and .bleurt) to one folder

5) Run `code/helper.py/report_metrics`

Table below describes evaluation results:

side constraint	***no-label***	***V-label***	***T-label***	***BLEU***	***BLEURT***
(reference file)	2078	3829	3777	-	-
(base-model)	2487	6428	767	14.78	**52.58**
none	6743	2634	305	11.36	45.70
informal	1351	150	8181	12.57	47.37
polite	603	9079	0	12.02	46.99
oracle	2213	3952	3517	**15.35**	49.96

All T/V labelling evaluation was conducted using the grammar-based detector.

BLEURT scores are multiplied by 100, originally it is scaled from 0 to 1.

(reference file) - Russian gold translation file

(base-model) - model trained without T/V labelling, all other scores are for T/V aware model

none - no labels prepended to test sentences

informal - all test sentences with <T> prefix

polite - all test sentences with <V> prefix

oracle - all test sentences are labeled with annotations of reference Russian translation

We see, that after training neural model connects <T> and <V> tokens with honorifics distinction. Forcibly pushing it to V-mode results in 0 T-sentences. Providing oracle labels produces better BLEU and BLEURT scores (+4 points). However, model without T/V labels in training data performs better in terms of BLEURT scores (+2.62). In my opinion, it can mean that learning additional features like T/V distinction can actually hurt generalisation ability of the model.

Conclusion

Machine translation should not only produce semantically accurate translations, but should also consider pragmatic aspects, such as producing socially appropriate forms of address. In this project, I confirm results obtained by Sennrich et. al. (2016) for different language pairs. By annotating the T-V distinction in the target text, and integrating the annotation as an additional input during the training of a neural translation model, we can apply side constraints at test time to control the production of honorifics in NMT.

References

1. Controlling Politeness in Neural Machine Translation via Side Constraints (Sennrich et al., NAACL 2016)
2. When a Good Translation is Wrong in Context: Context-Aware Machine Translation Improves on Deixis, Ellipsis, and Lexical Cohesion (Voita et al., ACL 2019)
3. Joey NMT: A Minimalist NMT Toolkit for Novices (Kreutzer et al., EMNLP 2019)
4. BLEURT: Learning Robust Metrics for Text Generation (Sellam et al., ACL 2020)