Machine Translation in Practice for PV061

FI:PV061: Introduction to MT
Michal Štefánik
stefanik.m@mail.muni.cz

Outline

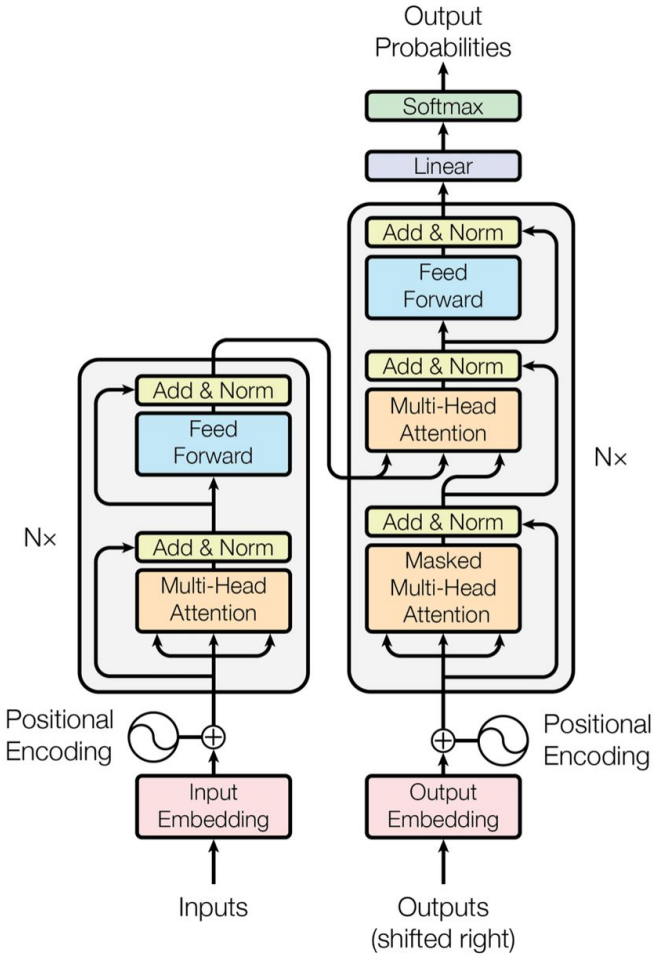
1. Motivation
2. Background
3. Practical problems
4. {Pre/Post}processing
5. Generation heuristics
6. Deployment

Motivation

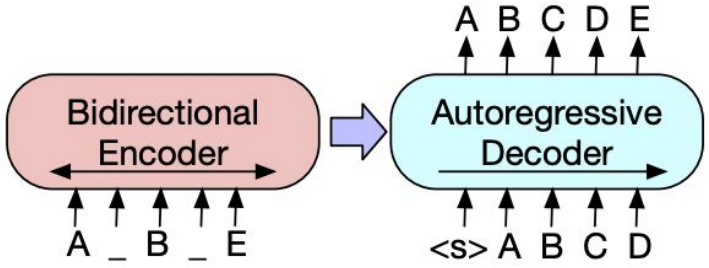
What is wrong about just using Google Translate?

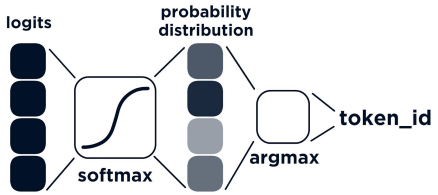
- Price
- Speed
- Robustness on specific domains

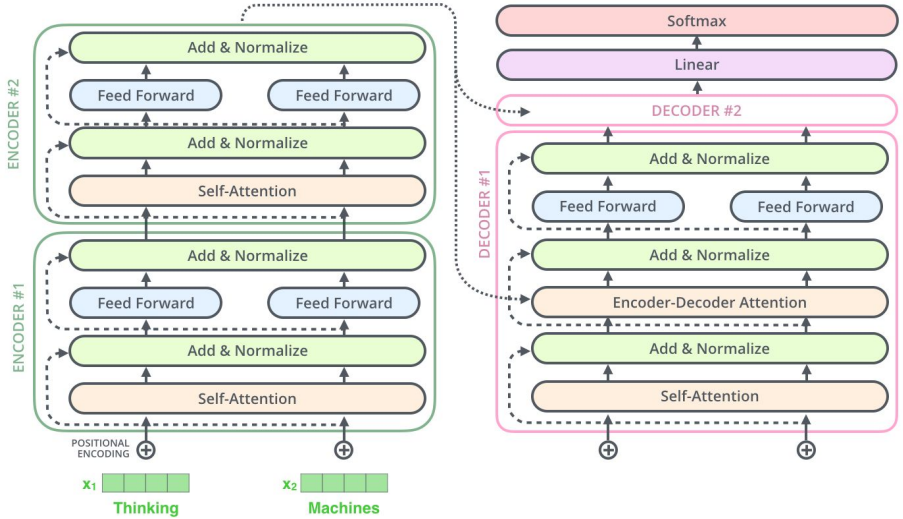
Background

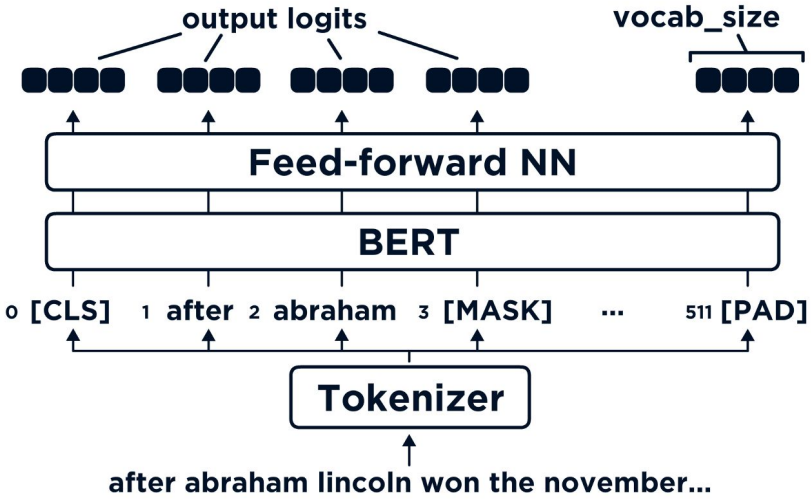

Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Nx
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Nx
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

Background - training

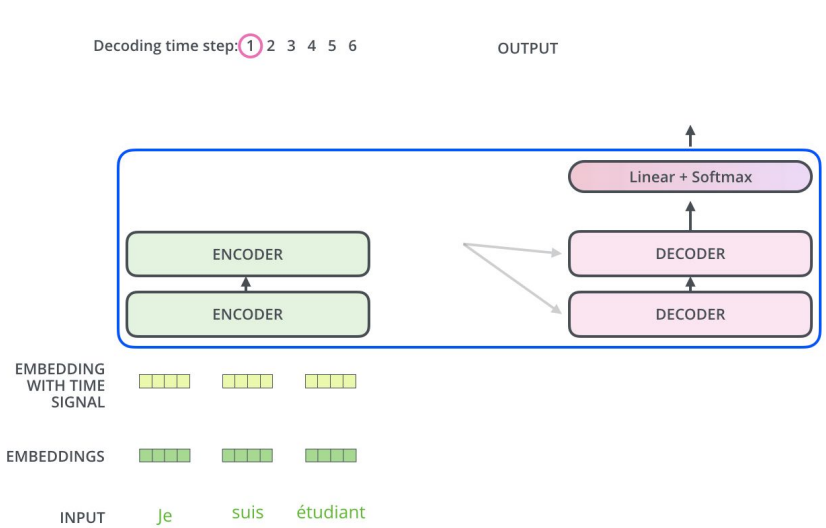


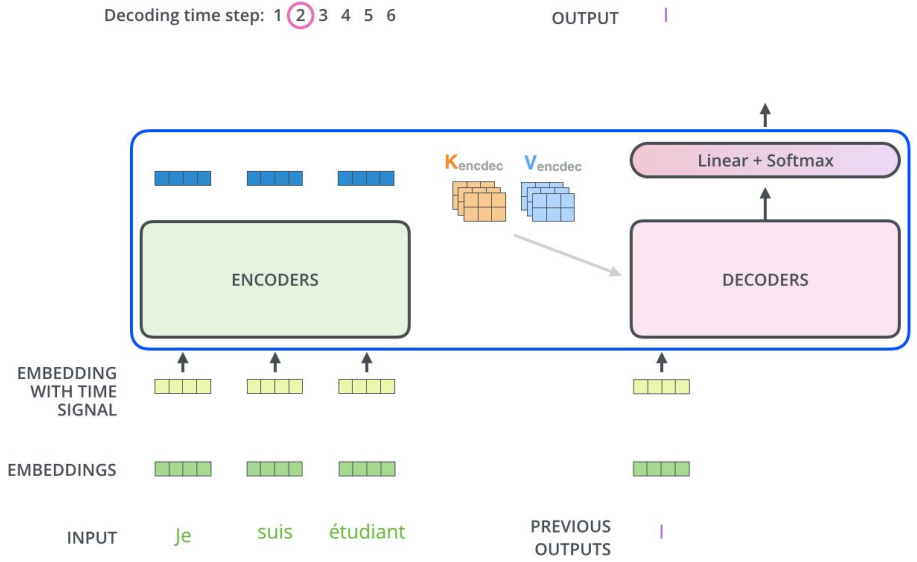






Background - inference





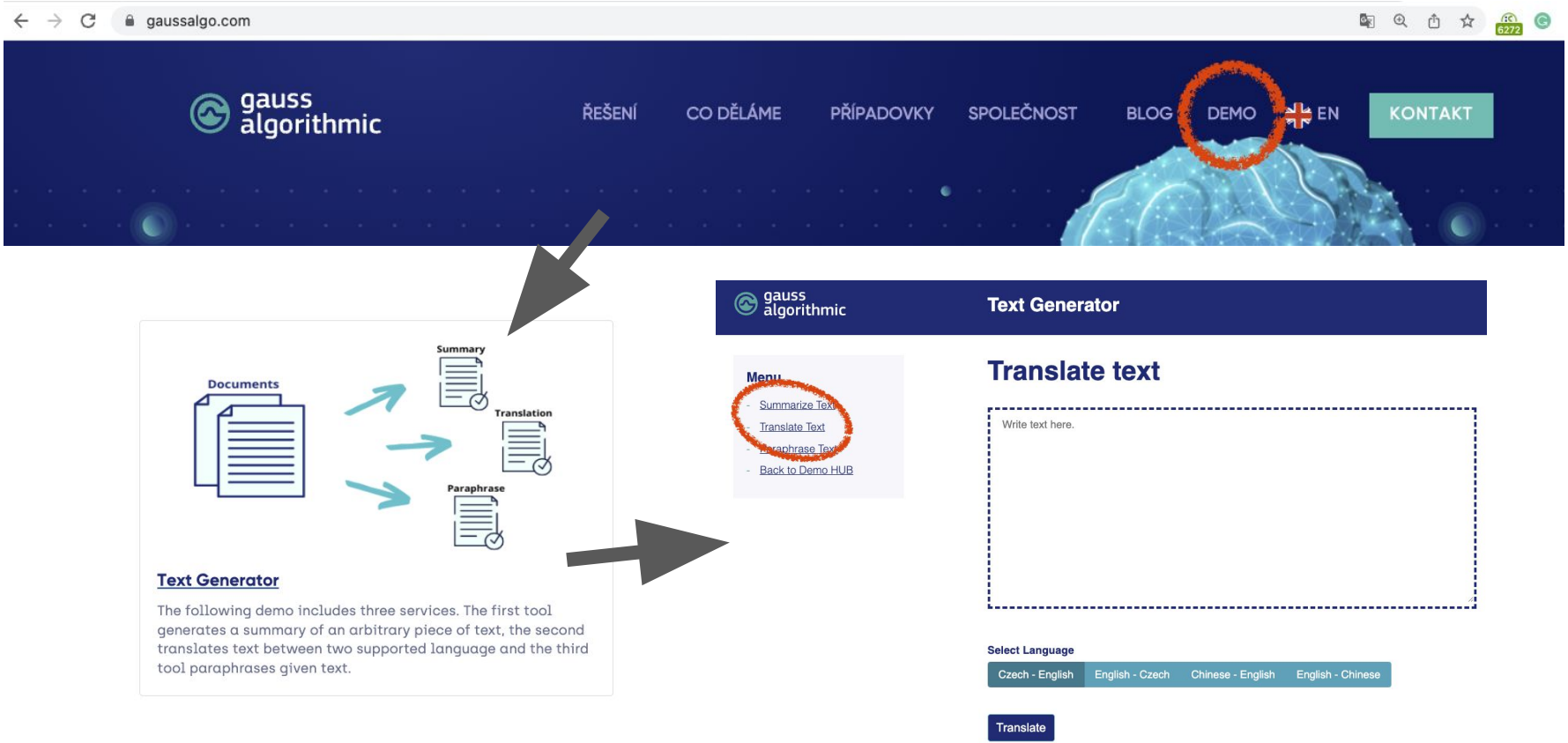
Practical problems

- What if we have specific vocabulary for some terms?
- What if the translator never seen some of the symbols?
- What if the text is non-canonical (i.e. weird)?

Our approach

- We need to serve 10+ language pairs, so we choose a big, pre-trained model and fine-tune it for our purposes (mBART)
- We take special care of the **non-pretrained languages** - we use **auxiliary language** if it is low-resource (like zh_TW)
- We train it to natively support the {pre/post}-processing that we need (later)

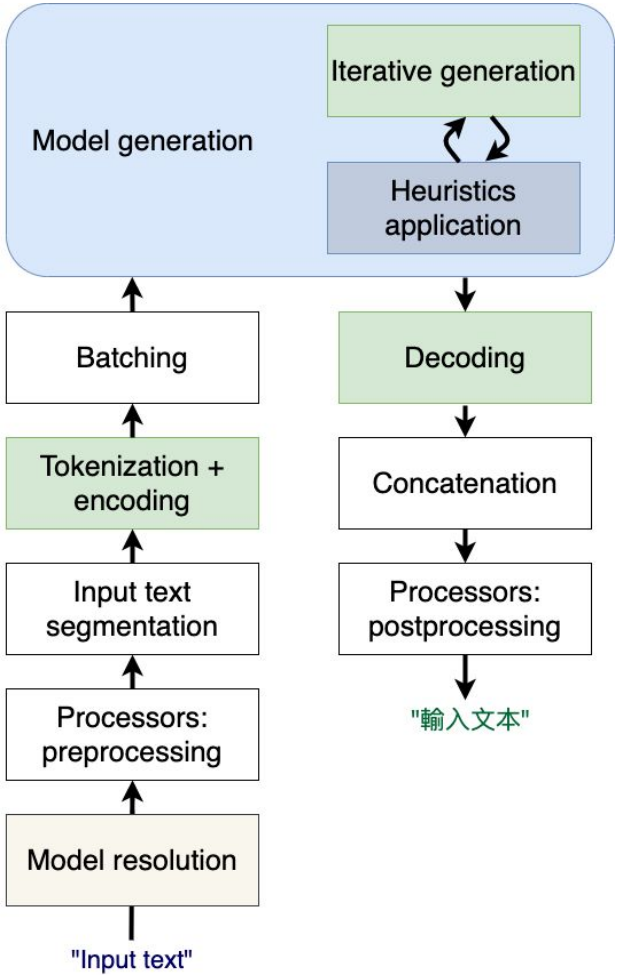
Demo


gaussalgo.com
gauss algorithmic
ŘEŠENÍ
CO DĚLÁME
PŘÍPADOVKY
SPOLEČNOST
BLOG
DEMO
EN
KONTAKT
Documents
Summary
Translation
Paraphrase
Text Generator
The following demo includes three services. The first tool generates a summary of an arbitrary piece of text, the second translates text between two supported language and the third tool paraphrases given text.
gauss algorithmic
Text Generator
Menu
Summarize Text
Translate Text
Paraphrase Text
Back to Demo HUB
Translate text
Write text here.
Select Language
Czech - English
English - Czech
Chinese - English
English - Chinese
Translate

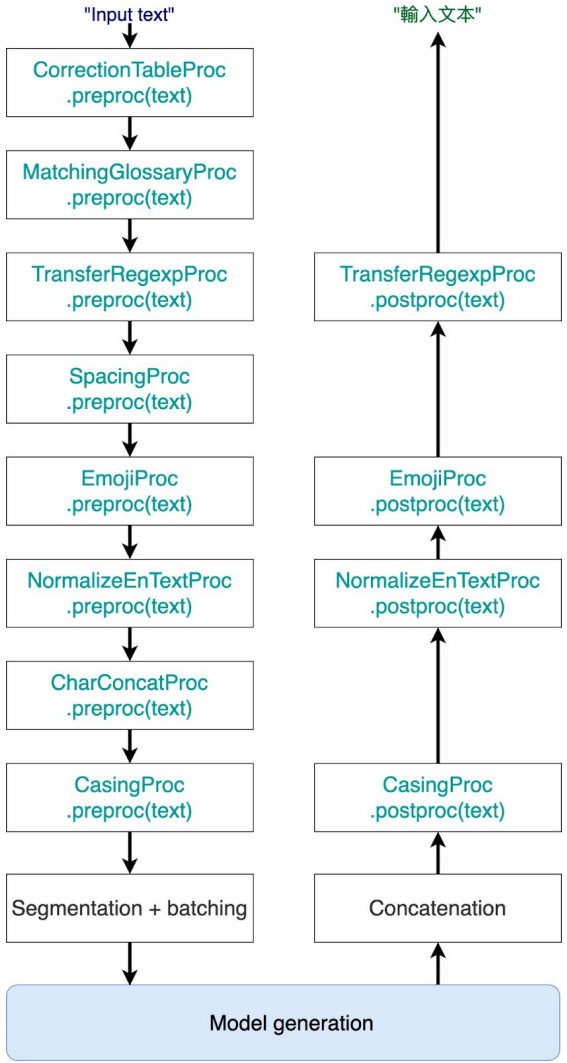
Demo

```
curl text-translator-api.gaussalgo.com/translate/ \
        -X POST \
        -H 'content-type: application/json' \
        -d '{"source_lang":"en_XX", "target_lang":"cs_CZ", "text": "Weird text to break the demo"}'
```

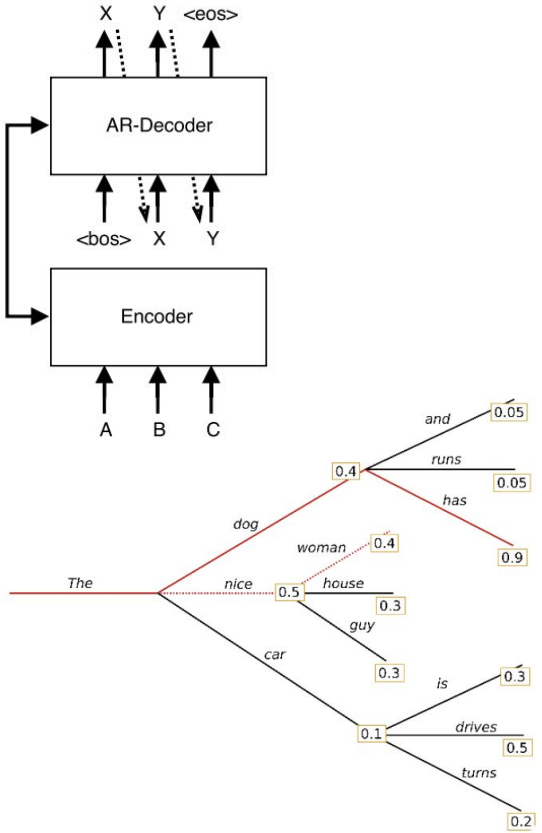
Application overview



{Pre/Post}processing



Generation heuristics



Our translator applies these heuristics:

1. **NoBadWords:** translations containing tokens from the list for the selected languages (such as Chinese or Arabic in Indonesian and some shared tokens) will get manually-assigned score of -infinity
2. **RepetitionPenalty:** scores of tokens that were already generated is multiplied by DEFAULT_REPETITION_PENALTY, hence lowered (logits are negative)
3. **MinLength:** Sequences of logits shorter than given threshold are set to -inf. This helps us to avoid the early generation termination.
4. **ForcedBeginningOfSequenceToken:** all the sequences not starting with given language token are assigned -inf. This is a support for mBART discourse interface.
5. **ForcedEndOfSequenceToken:** if some candidate sequence already contains <\s>, all the others are set to -inf, hence pruned and so the generation process ends. This is a speed-up trick that allows the generation to stop as soon as possible.

Deployment

- Training of mBART on batch_size=1 consumes 20GB of GPU, our current mBART wa fine-tuned on a single A100 for ~90 hours (~350USD) (link to the training tracker)
- Production uses kubernetes engine management
    - We fit three instances (models) to a single node with Nvidia Tesla T4 (~700USD/month)
    - Performance equivalent to 64-core CPUs (~1500USD/month)
    - Auto-scaling
- Each customer gets their own configurations of
    - Manual translation vocabulary
    - Abbreviations
    - Processors
    - Frequent typo corrections

- Currently, we manage to share the same model among all customers but this will soon change

Thanks!

Michal Štefánik

stefanik.m@mail.muni.cz