University of Tartu

Robotics and Computer Engineering

Institute of Technology

Speech based English-Estonian Neural Machine Translation

Md Rezwanul Islam

Master's Thesis in Computer Engineering (30 ECTS)

Supervisors:

Assoc. Prof. Gholamreza Anbarjafari (Shahab)
Hasan Sait Arslan

Tartu 2019

Resümee/Abstract

Keelepõhine Inglise-Eesti nervivõrgul rajanev masint— olge

Võrreldes teisi Euroopa keelte masinõppel rajanevaid tõlkeid Inglise-Eesti tõlgetega, naeme suurt lahknevust. Minu huvi tehisintellekti ja narvivõrkude vastu viis mind narvivõrgul põhineva Inglise-Eesti masintõlke rajamiseni. Toos kontrollitakse taiendavalt kõnetuvastus, gTTS, Torch, OpenNMT-py and SentencePiecei toovõimet, et narvivõrku oleks võimalik koostada mitte ainult teksti, vaid ka kõne kujul.

Masinõppel põhinev tõlge on tehisintellektiga seotud teemade hulgas uks raskemaid. Selle efektiivseks tootamiseks on loodud mitmeid raamistikke, algoritme ja toovahendeid. Selles projektis on kasutusel uks konkreetne raamistik, mis on valja toodud WMT17s. Selle peamine suundumus on transformatsiooni mudeli arendamine.

Kaesolev masin— oppel põhinev tõlge tootab tõhusalt konkreetsetes tingimustes ja naitab BLEU tulemuseks 40˜60. See on muljetavaldav. Siiski, minu eesmark ei olnud toimiva tõlkeprogrammi loomine ega BLEU skoori tõstmine, vaid ma soovisin valja tootada kõnel põhineva Inglise-Eesti automaatse tõlkija, mida oleks lihtne kasutada, mis oleks modulaarne ja mis kasutaks Attention Transformer raamistikku.

CERCS: P176 Articial intelligence [36]

Märksõnad: Neural machine translation(NMT), attention mechanism, speech recognition, speech synthesis, OPUS, SentencePiece, Torch, OpenNMT-py.

Speech based English-Estonian Neural Machine Translation

When comparing other European languages with 'English-Estonian' in the field of machine translation, we see a huge gap. My motivation to study Artificial Intelligence and Neural Network drove me to build a neural network for English to Estonian machine translation. Additionally checking the efficiency of SpeechRecognition, gTTS, Torch, OpenNMT-py and SentencePiece to build this not only for text but also for speech.

Machine translation is among hard problem in the field of AI. There are lot of architecture, algorithm and tools been developed to make it efficient. However, in this project I would follow a specific architecture proposed in WMT17 [1]. Where, developing a transformer model is the main idea to implement a speech based translation unit.

This machine translation performs efficiently in limited conditions and show BLEU 40˜60.

Which is impressive. However, developing a well performed translation system or increasing the BLEU score were not the targets for this project rather I wanted to develop a speech based English-Estonian translator which is easy to implement, modular and support Attention Transformer architecture.

CERCS: P176 Articial intelligence [36])

Keywords: Neural machine translation(NMT), attention mechanism, speech recognition, speech synthesis, OPUS, SentencePiece, Torch, OpenNMT-py.

Contents

List of figures

List of tables

1 Introduction

'Machine translation (MT) is the application of computers to the task of translating texts from one natural language to another. One of the very earliest pursuits in computer science, MT has proved to be an elusive goal, but today a number of systems are available which produce output which, if not perfect, is of sufficient quality to be useful in a number of specific domains.' [2]

Neural Machine Translation (NMT) is already popular and achieved tremendous performance in large scale translation task for the common European languages like English-French, English-Spanish or English-German. NMT is popular because it is easy to implement, maintain and required very less knowledge on low level complexity. For the corpus instance, I used open source corpus instance from opus.nlpl.eu. Which is extensively collected from online free to use. All preprocessing was done automatically. This model is trained in and end to end model, so that it can translate long sentences efficiently. As I followed here machine translation technique instead of any statistical method, so did not require any large storage to store gigantic phrase table and language model. Hence, NMT has a small memory footprint.

Therefore, this speech translation method for English-Estonia language can be divided into three major parts. Speech recognition, Neural Machine Translation and Speech Synthesis. For speech recognition, there are already implemented API like google cloud API, wit.api, bing speech recognition, IBM speech to text. Few of them are open source like CMU sphinx [12]. In my project, I used CMU sphinx, because it is easy to use, higher accuracy rate and free. Finally, after successful text translation, a speech synthesizer been used to synthesize text into speech. For this part, I used 'Google Text to Speech (gTTS)' library for python.

1.1 Problem overview

Estonina language is a Uralic language of Finnic branch. Finnish, Hungarian and Estonian are most common Uralic languages. In general, a natural language translation from voice is challenging in different ways. It includes complexity of grammar and cultures. NMT show tremendous performances for this type of natural languages.

Since this is speech based machine translation, we can separate all procedures in three modules,

1. English speech to English text recognition.

2. English text to Estonian text translation.

3. Estonian text to Estonian speech synthesis.

Estonian language has comparatively complex grammar and I don't have sufficient Estonian language skill. So it's quite hard for me to develop a translator model using methods like SMT or RBMT. So here is my savior, I would use NMT, because it's easy to use, efficient and requires less or no language knowledge for developers. So far a machine translation system is a tool that helps purposefully. Still it is not a replacement for the older system of translation rather it's an enhancer.

Since, I would develop a speech based translation system it's relevant to study problems for speech recognition system and speech synthesis separately.

Speech recognition technology has advanced rapidly, particularly in the past 10 to 15 years and is becoming commonplace in the bedroom, kitchen, and the rest of your house. It is light-years beyond its first iterations in the 1990s. Google alone has reduced its speech-recognition error rate by 30 percent since 2012. Among all speech recognition system there are few big players like Apple, Amazon and Microsoft developed AI based system on their platform. Most of those systems has problems in recognizing unfamiliar words or sentences with different unfamiliar accent. That means speech recognition system works better for example US/UK English and common words. In this project I would use SpeechRecognition package for python which is modern and state of the art solution currently. Similarly for speech synthesis I used 'Google Text to Speech(gTTS)' library for python.

Compared to other European languages, there are just few works been done for English to Estonian language translation. Additionally I could not compare the result for existing machine translation system for the same language pair with my project.

1.2 Objective

In our daily life we already know about some devices and application which can translate speech from one language to another. Some very good examples are Google translate and few others mobile based app. There are lot of research been done to translate speech efficiently for common European languages, while researches are very limited for English-Estonian language.

We can observe that knowledge are spreading all over the world. Every nationality are trying to gather knowledge in different form particularly over Internet. The knowledge and information are also increasing not only in English. The motivation behind my thesis topic was to have hands on experience to built a modern Neural Machine Translation system. Which can

be interacted not only by text but also by speech. After implementing this I would have enough knowledge of core technologies of any machine translator system like google translator, neural network, speech recognition and speech synthesis.

Machine Translation (MT) is a highly interdisciplinary and multidisciplinary field as it is approached by human translators, engineers, computer scientists, mathematicians and linguists. Overall speech translation has great impact over knowledge transferring which drives to better life for everyone.

After completing this project we can unleash the technique of modern machine translation system for everyone. Moreover, can discuss the advantages of using attention based Neural Machine Translation(NMT) over other NMT and SMT procedures. Besides, how to use latest libraries and technique for speech recognition, speech synthesis and neural network to develop a speech based English-Estonian translation system.

2 Related Works

In this section I will discuss about related works on voice based machine translation. Every voice based machine translation has 3 main independent units. That's why it is important to evaluate related works separately.

In recent years, Neural Machine Translation (NMT) based translation improved the overall translation technique and number of research in these particular area. One of the main advantages of using NMT over statistical method is it requires minimum domain knowledge. That's why we can see lot of researchers emphasis on machine translation to translate a natural language.

Neural Machine Translation is an approach to machine translation that uses a large artificial neural network to predict the likelihood of a sequence of words, typically modeling entire sentences in a single integrated model. One of the earliest goals for computers was the automatic translation of text from one language to another.

Automatic or machine translation is perhaps one of the most challenging artificial intelligence tasks given the fluidity of human language. Classically, rule-based systems were used for this task, which were replaced in the 1990s with statistical methods. More recently, deep neural network models achieve state-of-the-art results in a field that is aptly named neural machine translation.

For a long time (1950 – 1980), machine translation was done through the study of the linguistic information about the source and target languages, generating translations based on the dictionaries and grammars, which is called rule-based machine translation (RBMT). With the development of Statistics, statistical models started to be applied to machine translation, which generates translations based on the analysis of bilingual text corpus. This method is known as the statistical machine translation (SMT), which gained better performance than RBMT and dominated the field from the 1980s to 2000s. In the year of 1997, Ramon Neco and Mikel Forcada came up with the idea of using encoder-decoder structure to do machine translations [24]. A few years later in 2003, a group of researchers at the University of Montreal led by Yoshua Bengio developed a language model based on neural networks [25], which improved the data sparsity problem of traditional SMT models. Their work laid a foundation for the future usage of neural networks on machine translation.

Speech recognition technology is used to convert speech to text automatically. For a voice based translation system speech recognition system is necessary. The story of speech recognition is as much about the application of different approaches as the development of raw technology, though the two are inextricably linked. Over a period of decades, researchers would conceive of myriad ways to dissect language: by sounds, by structure and with statistics.

It wasnt until the mid 20th century that our forebears built something recognizable as ASR. Among the earliest projects was a digit recognizer called Audrey, created by researchers at Bell Laboratories in 1952. In the 1960s, IBM developed Shoe-box, a system that could recognize digits and arithmetic commands like plus and total. Meanwhile researchers in Japan built hardware that could recognize the constituent parts of speech like vowels; other systems could evaluate the structure of speech to figure out where a word might end.

Thankfully there was more optimism elsewhere. In the early 1970s, the U.S. Department of Defenses ARPA (the agency now known as DARPA) funded a five-year program called Speech Understanding Research. This led to the creation of several new ASR systems, the most successful of which was Carnegie Mellon Universitys Harpy, which could recognize just over 1000 words by 1976. In this table below there is list of speech recognition system in modern days.

Speech synthesis is the process of generating spoken language by machine on the basis of written input. Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech computer or speech synthesizer, and can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech.

Pytsx is a cross-platform text-to-speech wrapper. It uses different speech engines based on our operating system:

nsss - NSSpeechSynthesizer on Mac OS X 10.5 and higher

sapi5 - SAPI5 on Windows XP, Windows Vista, and (untested) Windows 7

espeak - eSpeak on any distro / platform that can host the shared library (e.g., Ubuntu / Fedora Linux)

There are few other commercial text to speech synthesis library which are targetting different framework and programming language.

Below is a table to show the available speech recognition system.

TABLE 1

Technologies for speech recognition.

Technologies	Type	Recognition type	Vocabulary size allowed	Common usage	Recognition quality	Default support languages	New languages allowed	Speech synthesis supported	Free or open source
Microsoft Speech API	Windows COM API	Dictation, complex phrase recognition	Large	Desktop application development	High	Few	No	Yes	No
Microsoft .NET System. Speech namespace	Windows .NET API	Dictation, complex phrase recognition	Large	Desktop application development	High	Few	No	Yes	No
Microsoft Speech Platform	Windows API	Commands, isolated word recognition	Large	Server application development	n/a	Many	No	Yes	No
Microsoft Unified Communications API	Windows API	Commands, isolated word recognition	Large	Server application development	n/a	Many	No	Yes	No
Sphinx 4	Complete Framework for SR	Dictation, complex phrase recognition	Large; depends on implementation	Speech recognition research studies	Depends on implementation	Few	Yes	No	Yes
HTK	Complete Framework for SR	Dictation, complex phrase recognition	Large; depends on implementation	Speech recognition research studies	Depends on implementation	None	Yes	No	No; license might be needed
Julius	Decoder system	Dictation, complex phrase recognition	Large	Speech recognition research studies	Depends on implementation	None	No	No	Yes
Java Speech API	Specification	n/a	n/a	n/a	Depends on implementation	None	No	Depends on implementation	Yes
Google Web Speech API	JavaScript API for Chrome	Dictation, complex phrase recognition	Large	Web application development	Average	Many	No	Not yet	Yes
Nuance Dragon SDK	API for desktop and mobile application	Dictation, complex phrase recognition	Large	Client, server and mobile application development	High	Many	No	Yes	No

Tables 2.1: Technologies for speech recognition

3 The Proposed System

To develop a voice-based machine translation system every system would have 3 units arranged one after another.

1. A speech to text speech recognition system.
2. A neural network to translate source text into target text.
3. A speech synthesizer to convert text into speech.

Here I developed a pipeline consisting all 3 units which can translate an English sentence into Estonian. The input and output both are voice based. In this part I will discuss both about working process and algorithm behind it.

The source code is there in GitHub [3].

3.1 Attention Mechanism Architecture

The major unit of a translation system is its translator mechanism. Here I would describe the necessary steps and algorithm behind this. At first, I would develop a neural network using attention mechanism and mostly depend upon an interesting paper titles, Attention All You Need published in 2017 [1]. The model architecture is a lot improvement for soft attention and make it possible to do seq2seq modeling without recurrent network units. The proposed transformer model is entirely built on the self-attention mechanisms without using sequence-aligned recurrent architecture.

We pay attention on any speech or image to find any corelated words or specific region of image. Human speech attention allows us to focus on some certain part of speech and based on this it adjusts the words on both sides. Suppose, we can explain the relationship between words in one sentence or close context. When we see eating, we expect to encounter a food word very soon. The color term describes the food, but probably not so much with eating directly.

More or less, attention in the profound learning can be comprehensively deciphered as a vector of significance loads: so as to anticipate or derive one component, for example, a pixel in a picture or a word in a sentence, we estimate utilizing the attention vector how firmly it is corresponded with different components and take the total of their qualities weighted by the attention vector as the approximation of the target.

'Attention is All you Need' (Vaswani, et al., 2017), without a doubt, is one of the most effective and fascinating paper in 2017. It exhibited a ton of enhancements to the soft attention and make it possible to do seq2seq modeling without recurrent network units. The proposed 'transformer' model is entirely built on the self-attention mechanisms without using sequence-aligned recurrent architecture.

The secret formula is conveyed in its model architecture.

Key, Value and Query

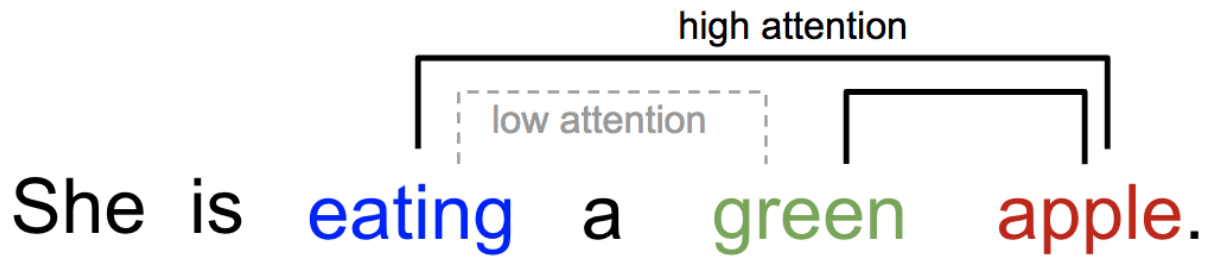
The significant part in the transformer is the unit of multi-head self-attention mechanism. The transformer views the encoded representation of the input as a set of key-value pairs, (K, V), both of dimension n (input sequence length); with regards to NMT, both the keys and values are the encoder hidden states. In the decoder, the previous output is compressed into a query (Q of dimension m) and the next output is produced by mapping this query and the set of keys and values. The transformer adopts the scaled dot-product attention: the output is a weighted sum of the values, where the weight assigned to each value is determined by the dot-product of the query with all the keys:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{n}})V$$

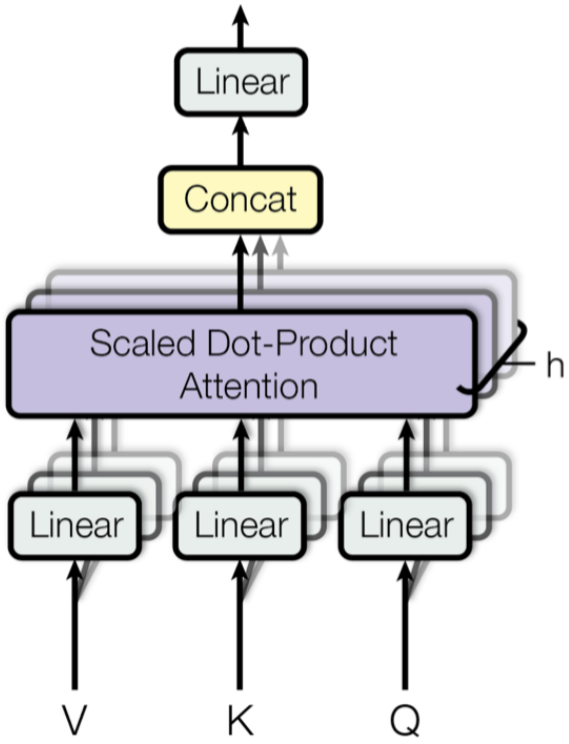
Multi-Head Self Attention with Encoder and Decoder

Instead of just processing the attention once, the multi-head mechanism runs through the scaled dot-product attention multiple times in parallel. The independent attention outputs are simply concatenated and linearly transformed into the expected dimensions. According to the paper, "multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. With a single attention head, averaging inhibits this."

$$MultiHead(Q, K, V) = [head_1; ..head_h]W^0$$

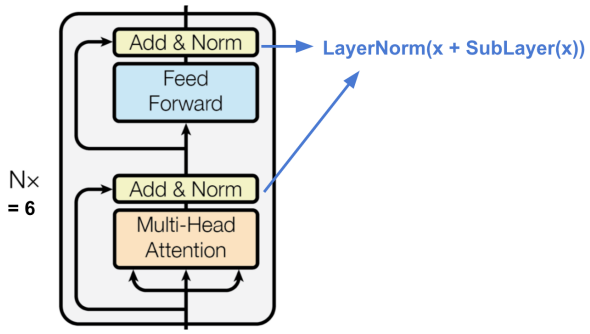


Figures 3.1: One word 'attends' to other words in the same sentence differently. Source: [28]



Figures 3.2: Multi-head scaled dot-product attention mechanism. (Image source: Fig 2 in [1])

$$Where, \quad head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$$



Figures 3.3: The transformer's encoder. Image source: [1]

The encoder generates an attention-based representation with ability to find a particular snippet of data from a possibly vast context.

1. A stack of N = 6 identical layers.

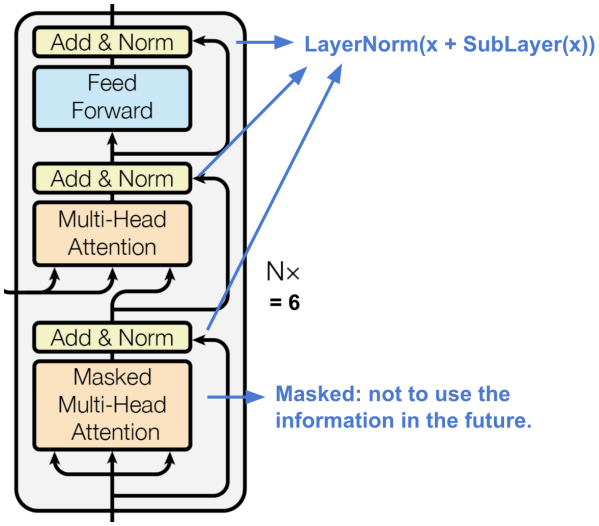
2. Each layer has a **multi-head self-attention layer** and a simple position-wise **fully connected feed-forward network**.

3. Each sub-layer adopts a residual [29] connection and a layer **normalization**. All the

sub-layers output data of the same dimension,

$$d_{model} = 512$$

.



Figures 3.4: The transformer's decoder. Image source: [1]

The decoder is able to retrieval from the encoded representation.

1. A stack of N = 6 identical layers.
2. Each layer has two sub-layers of multi-head attention mechanisms and one sub-layer of fully-connected feed-forward network.
3. Similar to the encoder, each sub-layer adopts a residual connection and a layer normalization.
4. The first multi-head attention sub-layer is modified to prevent positions from attending to subsequent positions, as we dont want to look into the future of the target sequence when predicting the current position.

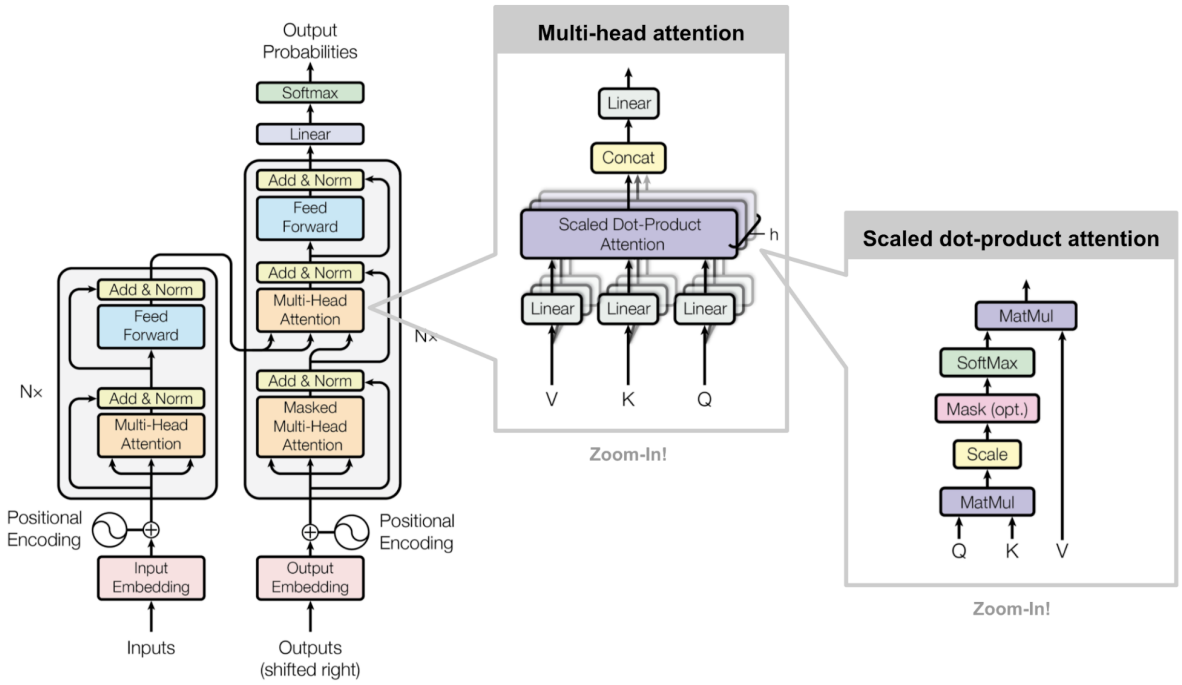
Finally here is the complete view of the transformers architecture:

1. Both the source and target sequences first go through embedding layers to produce data of the same dimension

$$d_{model} = 512$$

.

2. To preserve the position information, a sinusoid-wave-based positional encoding is applied and summed with the embedding output.



Figures 3.5: The full model architecture of the transformer. Image source: Fig 1 & 2 in [1]

3. A softmax and linear layer are added to the final decoder output.

Following this attention mechanism, I would develop a English to Estonian translator model using different tools like SentencePiece and OpenNMT-py. Finally, this translator model would help us to translate a English text into Estonian.

3.2 Collecting English-Estonian Parallel Corpus

Parallel corpora are central to translation studies and contrastive linguistics [27]. Many of the parallel corpora are accessible through easy-to-use concordances which considerably facilitates the study of interlinguistic phenomena. Such corpora are also a rich source of materials for language teaching. Furthermore, parallel corpora serve as training data for statistical machine translation systems.

To develop a similar type of translation system for any language we can collect corresponding parallel corpus from Open Source Parallel Corpus (OPUS) [5]. Definitely, I choose English to Estonian from the dropdown and downloaded the untokenized raw files [OpenSubtitles v2016]. OPUS is a growing collection of parallel corpora on internet. Its not targeting any specific domain. Domain specific parallel corpus might not be useful for general purpose translation, which I described in result and discussion section in detail.

In the OPUS project contributors try to convert and align free online data, to add linguistic annotation, and to provide the community with a publicly available parallel corpus. OPUS is based on open source products and the corpus is also delivered as an open content package. They

Search & download resources: en (English) ▾ et (Estonian) ▾ all ▾

Language resources: click on [tmx | moses | xces | lang-id] to download the data! (raw = untokenized, ud = parsed with universal dependencies, alg = word alignments and phrase tables)

corpus	doc's	sent's	en tokens	et tokens	XCES/XML	raw	TMX	Moses	mono	raw	ud	alg	dic	freq			other files
OpenSubtitles v2018	15169	13.3M	104.3M	82.5M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et	query	sample	xces/alt
OpenSubtitles v2016	12222	11.2M	86.5M	69.0M	xces en et	en et	tmx	moses	en et	en et		alg	dic	en et		sample	
DGT v4	26599	3.1M	71.1M	43.7M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et	query	sample	
OpenSubtitles v2012	7185	6.9M	53.9M	42.9M	xces en et	en et				en et				en et		sample	
OpenSubtitles v2013	5938	6.0M	47.4M	37.1M	xces en et	en et				en et				en et		sample	
OpenSubtitles v2011	4828	4.5M	35.0M	28.0M	xces en et	en et										sample	
JRC-Acquis v3.0	11945	0.8M	33.6M	26.1M	xces		tmx	moses	en et	en et				en et		sample	
ParaCrawl v1	13	1.3M	20.8M	19.7M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et	query	sample	
Europarl v7	8101	0.7M	17.6M	13.1M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et	query	sample	
EUbookshop v2	1140	0.4M	17.7M	12.5M	xces en et	en et	tmx	moses	en et	en et			dic	en et	query	sample	moses/strict
EMEA v3	1874	1.1M	11.7M	12.2M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et	query	sample	
ECB v1	1	0.1M	3.2M	2.3M	xces en et	en et	tmx	moses	en et	en et		alg		en et	query	sample	
KDE4 v2	2132	0.3M	2.4M	1.9M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et	query	sample	
Tatoeba v2	1	0.7k	3.6M	6.3k	xces en et	en et	tmx	moses	en et	en et		alg smt		en et	query	sample	
GNOME v1	1106	0.4M	2.0M	1.2M	xces en et	en et	tmx	moses	en et	en et		alg smt		en et		sample	
Ubuntu v14.10	313	68.0k	0.4M	0.2M	xces en et	en et	tmx	moses	en et	en et		alg smt	dic	en et		sample	
OpenSubtitles v1	29	33.1k	0.3M	0.2M	xces en et	en et	tmx	moses	en et	en et		alg smt		en et	query	sample	
EUconst v1	47	10.2k	0.2M	0.1M	xces en et	en et	tmx	moses	en et	en et		alg		en et	query	sample	
Europarl v3					xces en	en			en	en				en	query		
total	**98643**	**50.1M**	**511.8M**	**392.8M**	**50.1M**		**32.7M**	**32.7M**									

Figures 3.6: English to Estonian corpus from OPUS.

used several tools to compile the current collection. All pre-processing is done automatically. No manual corrections have been carried out.

In .tgz file all parallel corpus are in different files. We have to concat all together. For source sentences it can be 'en.txt' and for target sentence 'et.txt'. Now we have final parallel corpus which contains 10000 English and

3.3 Training a Translation Model Using OpenNMT-py

A translation model is required to translate a source text into target text. In this section we will develop a translation model using OpenNMT-py and attention mechanism. OpenNMT-py is a PyTorch port. It is open source neural machine translation system which is research friendly, easy to use and efficient for machine translation, image-to-text, morphology and many other domains.

OpenNMT-py or Open-NMT is based on the research by Guillaume Klein [7]

As per the Paper, the following details are discovered about its architecture:

" OpenNMT is a complete library for training and deploying neural machine translation models. The system is successor to seq2seq-attn developed at Harvard, and has been completely rewritten for ease of efficiency, readability, and generalizability. It includes vanilla NMT models along with support for attention, gating, stacking, input feeding, regularization, beam search and all other options necessary for state-of-the-art performance.

The main system is implemented in the Lua/Torch mathematical framework, and can be easily be extended using Torchs internal standard neural network components. It has also been extended by Adam Lerer of Facebook Research to support Python/PyTorch framework, with the same API. "

Installing Required Packages

In this part I will explain how to teach a neural network to translate English text into Estonian. I used OpenNMT-py to create a neural network. OpenNMT-py a pyTorch port of OpenNMT. At first we need to install OpenNMT-py following their installation procedures.

The priliminary step, of course is to clone the OpenNMT-py repository:

Listing 3.1: Clone OpenNMT-py from Github

```
1 git clone https://github.com/OpenNMT/OpenNMT-py
2 cd OpenNMT-py
```

Heres a requirements.txt file to gather all the required packages :

Since PyTorch is continuously evolving, it is recommended to fork the PyTorch 0.4 version to ensure a stable performance of the code base.

Then let's run the following command to automatically gather pre-requisite dependencies:

Listing 3.2: Installing all required packages for OpenNMT-py

```
1 pip install -r requirements.txt
```

Preparing The Dataset from Parallel Corpus

Pre-processing the Data

For prepossessing, now I have the following:

- en.train.enc : Training file containing English (Source Language) sentences
- et.train.enc : Training file containing Estonian (Target Language) sentences
- en.dev.enc : Validation data consisting of English (Source Language) sentences
- et.dev.enc : Validation data consisting of Estonian (Target Language) sentences

All of the above files were placed in 'dataset' folder.

Listing 3.3: Preprocess data using OpenNMT-py

```
1 python ../OpenNMT-py/preprocess.py
2 -train_src ../dataset/en.train.enc
3 -train_tgt ../dataset/et.train.enc
4 -valid_src ../dataset/en.dev.enc
5 -valid_tgt ../dataset/et.dev.enc
6 -save_data ../dataset/data
```

Training the Data

After preprocessing , the main command to train the network is the following. This script would save all trained model into 'models' directory. The main train command is quite simple. Minimally, it takes a data file and save file.

Listing 3.4: Train data using OpenNMT-py

```
1
2  python ../OpenNMT-py/train.py -data \
3      ../dataset/data -save_model ../models/ \
4      -layers 6 -rnn_size 512 -word_vec_size 512 \
5      -transformer_ff 2048 -heads 8 \
6      -encoder_type transformer -decoder_type transformer \
7      -position_encoding -train_steps 350836 \
8      -max_generator_batches 2 -dropout 0.1 \
9      -batch_size 4096 -batch_type tokens -normalization tokens \
10     -accum_count 2 -optim adam -adam_beta2 0.998 \
11     -decay_method noam -warmup_steps 8000 -learning_rate 2 \
12     -max_grad_norm 0 -param_init 0  -param_init_glorot \
13     -label_smoothing 0.1 -valid_steps 10000 \
14     -save_checkpoint_steps 10000 \
15     -world_size 1 -gpu_ranks 0
```

To understand all parameters used here, it is recommended to read OpenNMT-py documentation [30, 31]. I used to stop this training after _step_350000.pt.

I used Vedur Cluster [32] for this training. The model was trained for 10000 epochs on a NVIDIA GEFORCE 2GB GPU. Training on a CPU will require a very high computational cost, hence it is recommended to use a high end GPU for training the model with a large amount of data at a faster rate.

This is last part to train a model. But using this model we still can translate an English sentence into Estonian (not speech based).

Testing the Data with Sample English Text

Now, testing part. Model for translation is ready. This model is capable to translate a English sentence into Estonian. But source sentence need to be encoded using SentencePiece [33], target sentence also need to be decoded using same library. Then we would get an Estonian sentence finally.

Here is the command to translate English text into Estonian text. Encoding and decoding separately by another commands.

Listing 3.5: Test English-Estonian using OpenNMT-py

```
python ../OpenNMT-py/translate.py -model
./models/_step_350000.pt -src ./en.txt -output
./pred.txt -replace_unk -verbose
```

3.4 Using a Speech Recognition System

Speech recognition is the ability of a machine or program to detect any spoken voice or phrases. There are few efficient library and API for speech recognition. However, I used open source Speech Recognition library in Python. It is easy to implement and maintain. Can be integrated with other popular API like CMU Sphinx, Google API, Bing voice recognition or IBM speech text. But after speech recognition, I developed my own Machine Translation system, that's why It was not needed any third party integration at this point.

There are several speech recognition technologies. For this project I used Python SpeechRecognition library with google API. It performs very well and easy to use. This API converts speech (from Microphone) to text. Besides, this API also can convert audio file into text. However, which is out of this project's scope. here we will use only Microphone as input device.

Listing 3.6: Source code speech recognition

```
import speech_recognition as sr

try:
    while True:
        with sr.Microphone() as source:
            print('say something')
            audio = r.listen(source)
            print('---time over---')

        try:
            print('Text: ' + r.recognize_google(audio));
        except:
            pass
```

3.5 Using a Speech Synthesis System

Speech synthesis is artificial production of human voice from text to speech. A speech synthesizer system can be implemented in software or hardware. The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood clearly.

In this project I would use gTTS (Google Text-to-Speech) [34] library to synthesize Estonian text into voice. This is a easy to use python library. After converting text to speech it will write data into mp3 file format.

Here is a sample code to use gTTS,

Listing 3.7: Source code, text to speech synthesis with gTTS

```
1
2 from gtts import gTTS
3 tts = gTTS('Ma olen Rezwan. Ma elan Tartus.', lang='et')
4 tts.save('et.mp3')
```

Parameter, lang = et states that the language is in Estonian. Default value is 'en' [English]. Any Estonian sentence in English synthesizer would not perform well. So this case we also need to consider.

3.6 Integrating All Modules for a Speech Based Translation System

Now, all the required modules are ready to integrate together. We would drive all 3 modules through a pipeline one by one. The sequences would be, 1) Speech to text, 2) Text tokenize, Translate and detokenize, 3) Text to speech.

In my case, I didn't use any UI rather just run python script one after another using bash.

Finally, I would run this bash script and try to evaluate the result.

4 Experimental Results and Discussion

In this section, we present our experimental results on publicly available corpora used extensively as benchmarks for Neural Machine Translation systems: WMT14 English-to-Estonian (WMT EnEt). On these datasets, we benchmark NMT models with word-based, 13 character-based, and sentence-piece-based vocabularies. We also present the improved accuracy of our models after ne-tuning with RL and model assembling.

Our main objective with these datasets is to show the contributions of various components in our implementation, in particular the sentence-piece model, RL model renement, and model assembling. In addition to testing on publicly available corpora, we also test NMT on Googles translation production corpora, which are two to three decimal orders of magnitudes bigger than the WMT corpora for a given language pair. We compare the accuracy of our model against human accuracy and the best Phrase-Based Machine Translation (PBMT) production system for Google Translate. In all experiments, our models consist of 8 encoder layers and 8 decoder layers. (Since the bottom encoder layer is actually bi-directional, in total there are 9 logically distinct LSTM passes in the encoder.) The attention network is a simple feed forward network with one hidden layer with 1024 nodes. All of the models use 1024 LSTM nodes per encoder and decoder layers.

4.1 Experimental Results

Bilingual Evaluation Understudy (BLEU) is an algorithm for evaluating the quality of machine traslation. Translation quality is considered to be the correspondence between a machine's output and that of a human: "the closer a machine translation is to a professional human translation, the better it is" this is the central idea behind BLEU [35].

In this project, I would evaluate the BLEU score using a Perl script which is integrated OpenNMT-py. Here is the perl script to evaluate score,

Listing 4.1: Evaluating the BLEU score

```
1
2 perl ../OpenNMT-py/tools/multi-bleu.perl
3 pred.txt < human.pred.atok
```

Here ‘human.pred.atok’ file contains tokenized translated sentences by human. In this case I took help from a native Estonian who translated all my source sentences into Estonian. Next step is to tokenize all sentences by SentencePiece and save into the file above.

Here is the BLEU score evaluation,

1. Sample source (en): “I am a person who is positive about every aspect of life. There are many things I like to do, to see, and to experience.”

BLEU = 44.08, 60.0/47.7/39.5/33.3 (BP=1.000, ratio=1.047, hyp_len=45, ref_len=43)

4.2 Discussion

Currently, in the field of machine translation, NMT system is hard to beat. The future looks promising and it will rule this domain. Philipp Koehn and Rebecca Knowles discussed [26] about the six problems in NMT. It includes domain mismatch, amount of training data, rare words, long sentences, word alignment and beam search. According to their researches, NMT has following types of difficulties,

NMT has lower out of domain capability, that means it can work well in specific domain based upon the parallel corpus. If we train on neural network based general day to day communication or from movie script, then this system might fail to translate a sentence in IT or Law domain.

System ↓	Law		Medical		IT		Koran		Subtitles	
All Data	30.5	32.8	45.1	42.2	35.3	44.7	17.9	17.9	26.4	20.8
Law	31.1	34.4	12.1	18.2	3.5	6.9	1.3	2.2	2.8	6.0
Medical	3.9	10.2	39.4	43.5	2.0	8.5	0.6	2.0	1.4	5.8
IT	1.9	3.7	6.5	5.3	42.1	39.8	1.8	1.6	3.9	4.7
Koran	0.4	1.8	0.0	2.1	0.0	2.3	15.9	18.8	1.0	5.5
Subtitles	7.0	9.9	9.3	17.8	9.2	13.6	9.0	8.4	25.9	22.1

Figures 4.1: Quality of systems (BLEU), when trained on one domain (rows) and tested on another domain (columns). Comparably, NMT systems (left bars) show more degraded performance out of domain. Source: [26]

NMT systems have a steeper learning curve with respect to the **amount of training data**. Thats in low resources, shows lesser performance and accuracy.

NMT systems that operate at the sub-word level (e.g. with byte-pair encoding) perform better than SMT systems on extremely **low frequency words**.

NMT systems have lower translation quality on very **long sentences** but do comparably better up to a sentence length of about 60 words.

The attention model for NMT does not always fulfill the role of a **word alignment model** but may in fact dramatically diverge.

Beam search decoding only improves translation quality for narrow beams and deteriorates when exposed to a larger search space.

However, this list is incomplete, and researchers are trying to analyze accurate the overall performance NMT over SMT on different natural language all over the world. Certainly, we can say that attention based encode-decoder system is more accurate for machine translation over any other techniques like convolution neural network (CNN) or recurrent neural network (RNN).

On the other hand, in this project I used SpeechRecognition library which is very efficient and shows very good results. We can configure this library for other languages besides English as well. There are few scopes to improve like recognize speech accurately in a noisy condition and if sentence contains any rare words. Similarly, I used gTTS, which is state of the art text to speech library developed by Google for python. It has better control over different natural language and we also can use different type of voice pattern like male, female or kid voice.

5 Conclusion and Future Work

In this report, I wrote all details to develop a English to Estonian speech based translation system using neural network. It also includes the accuracy, robustness and complexity of this working process and algorithm. On the translation benchmark, the result was reasonable with training data and computing hardware.

When comparing this neural machine translation with human translation, it gives 60% accuracy. Definitely, this accuracy would improve if we use larger data set for training with sentences contain rare words. Also using, efficient computing machine is also important.

Here, we examine a number of challenges to neural machine translation and give empirical results on how well the technology currently holds up, compared to traditional statistical machine translation. We show that, despite its recent successes, neural machine translation still has to overcome various challenges, most notably performance out-of-domain and under low resource conditions.

What a lot of the problems have in common is that the neural translation models do not show robust behavior when confronted with conditions that differ signicantly from training conditions may it be due to limited exposure to training data, unusual input in case of out of domain test sentences or unlikely initial word choices in beam search. The solution to the problems may hence lie in a more general approach of training that steps outside optimizing single word predictions given perfectly matching prior sequences.

Another challenge that we do not examine empirically: neural machine translation systems are much less interpretable. The answer to the question of why the training data leads these systems to decide on specic word choices during decoding is buried in large matrices of real-numbered values. There is a clear need to develop better analytics for neural machine translation.

Acknowledgement

I would like to express my deep appriciation to my thesis supervisors, Hasan Sait Arslan and Assoc. Prof. Gholamreza Anbarjafari (Shahab).

I would like to thank The High Performance Computing Center and all my faculty members at University of Tartu.

In addition, my love for all the researchers in the field of machine translation and all online contributors. I appreciate all their hard work for open source tools, projects, libraries and knowledge base for the betterment of human life. You all are awesome!

Md Rezwanul Islam

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need. https://arxiv.org/pdf/1706.03762.pdf

[2] European Association for Machine Translation http://www.eamt.org/

[3] Source code for this project, https://github.com/rezwanshubh/translator_en_et.git

[4] Python Speech Recognition Library, https://pypi.org/project/SpeechRecognition/

[5] An open source parallel corpus, http://opus.nlpl.eu/

[6] Jrg Tiedemann, 2012. Parallel Data, Tools and Interfaces in OPUS. In Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC'2012). http://www.lrec-conf.org/proceedings/lrec2012/pdf/463_Paper.pdf

[7] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart and Alexander M. Rush. OpenNMT: Open-Source Toolkit for Neural Machine Translation. https://doi.org/10.18653/v1/P17-4012

[8] Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks.pdf

[9] Natural Language Processing, https://en.wikipedia.org/wiki/Natural-language_processing

[10] Text-to-Speech Synthesis of Estonian https://www.researchgate.net/publication/221479645_Text-to-speech_synthesis_of_estonian

[11] Albert Einstein. *Zur Elektrodynamik bewegter Körper*. (German) [*On the electrodynamics of moving bodies*]. Annalen der Physik, 322(10):891921, 1905.

[12] Python SpeechRecognition, https://pypi.python.org/pypi/SpeechRecognition/

[13] Text-to-Speech Synthesis for Estonian, https://metashare.ut.ee/repository/browse/text-to-speech-synthesis-for-estonian/dbec262258b711e2a6e4005056b4002418584c0383ec405ab89b64546488c0ea/

[14] Python Speech Recognition Library, https://github.com/Uberi/speech_recognition

[15] Chrisman, L. Learning recursive distributed representations for holistic computation. Connection Science 3, 4 (1991), 345366

[16] Brown, P., Cocke, J., Pietra, S. D., Pietra, V. D., Jelinek, F., Mercer, R., and Roossin, P. A statistical approach to language translation. In Proceedings of the 12th Conference on Computational Linguistics - Volume 1 (Stroudsburg, PA, USA, 1988), COLING 88, Association for Computational Linguistics, pp. 7176.

[17] Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roossin, P. S. A statistical approach to machine translation. Computational linguistics 16, 2 (1990), 7985.

[18] Brown, P. F., Pietra, V. J. D., Pietra, S. A. D., and Mercer, R. L. The mathematics of statistical machine translation: Parameter estimation. Comput. Linguist. 19, 2 (June 1993), 263311.

[19] Koehn, P., Och, F. J., and Marcu, D. Statistical phrase-based translation. In Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics (2003).

[20] Cho, K., van Merrienboer, B., Glehre, ., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Conference on Empirical Methods in Natural Language Processing (2014).

[21] Kalchbrenner, N., and Blunsom, P. Recurrent continuous translation models. In Conference on Empirical Methods in Natural Language Processing (2013).

[22] Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R. M., and Makhoul, J. Fast and robust neural network joint models for statistical machine translation. In ACL (1) (2014), Citeseer, pp. 13701380

[23] Mos Zhang— Technical Report and Analysis produced by Synced Lab. History and Frontier of the Neural Machine Translation https://medium.com/syncedreview/history-and-frontier-of-the-neural-machine-translation-dc981d25422d

[24] Neco, R. P., & Forcada, M. L. (1997, June). Asynchronous translations with recurrent neural nets. In Neural Networks, 1997., International Conference on (Vol. 4, pp. 25352540). IEEE.

[25] Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. Journal of machine learning research, 3(Feb), 11371155.

[26] Philipp Koehn, Rebecca Knowles. Six Challenges for Neural Machine Translation https://aclweb.org/anthology/W17-3204

[27] Parallel corpora in the CLARIN infrastructure https://www.clarin.eu/resource-families/parallel-corpora

[28] Lilian Weng on Jun 24, 2018 https://lilianweng.github.io/lil-log/2018/06/24/attention-attention.html#whats-wrong-with-seq2seq-model

[29] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun at at Microsoft Research Deep Residual Learning for Image Recognition https://arxiv.org/pdf/1512.03385.pdf

[30] OpenNMT-py full documentation. http://opennmt.net/OpenNMT-py/

[31] OpenNMT-py documentation to train data. http://opennmt.net/OpenNMT-py/

[32] Vedur cluster. University of Tartu https://hpc.ut.ee/en/vedur-cluster/

[33] SentencePiece, is an unsupervised text tokenizer and detokenizer. https://github.com/google/sentencepiece

[34] Documentation: Google speech-to-text https://gtts.readthedocs.io/en/latest/

[35] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation https://www.aclweb.org/anthology/P02-1040.pdf

[36] Common European Research Classification Scheme (CERCS) Teadusvaldkondade ja -erialade klassifikaator https://www.etis.ee/Portal/Classifiers/Details/d3717f7b-bec8-4cd9-8ea4-c89cd56ca46e (ETIS); PDF: https://wiki.ut.ee/download/attachments/16581162/Common%20European%20Research%20Classification%20Scheme.pdf

Non-exclusive license to reproduce thesis and make thesis public

I, Md Rezwanul Islam

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   "Speech Based English-Estonian Neural Machine Translation"

   Supervised by Assoc. Prof. Gholamreza Anbarjafari (Shahab) and Hasan Sait Arslan.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive license does not infringe other persons intellectual property rights or rights arising from the personal data protection legislation.

Md Rezwanul Islam
Tartu, **20.05.2019**