

Universität zu Köln
Philosophische Fakultät
Institut für Digital Humanities
Public Humanities Tools
Dozent: Dr. Jürgen Hermes
Wintersemester 2021/2022



– Handbook for using–

Deep Pavlovs Named Entity Recognition

Contents

❖	Introduction.....	3
❖	Installation Guide.....	5
➤	Install Anaconda:	5
➤	Install Deep Pavlov:	5
➤	Install NER – Model	7
➤	Download Data for the pretrained model	8
❖	Using the Model.....	9
➤	Usage without Script	9
➤	Usage with Script	10
▪	Before you run the model for the first time	10
▪	Running the Model	10
➤	File System	10
❖	Limitations & possible Additions	11
❖	Alternatives to Deep Pavlov	11
❖	Links for further Information.....	13

❖ Introduction

Named-entity recognition (NER) is a task in information extraction and refers to the automatic identification and classification of proper names in texts. A proper name is a sequence of words that describes a real entity, such as a place or a person. For our project, we chose the open-source conversational AI framework: [Deep Pavlov](#), partly because it has a multilingual model. That was an advantage for us because we wanted to be able to work with Dostoyevsky works both in German and Russian. However, there are also other frameworks that have advantages and disadvantages. We made an overview of a selected few alternatives on page 10.

Also [here's](#) a quick Demo of what Deep Pavlov can do.

➤ NER-Tags:

Deep Pavlov uses a system of different tags in its pre-trained models. You can always (re-) train a model to identify more or different Tags from these if you like.

The list of available tags and their descriptions are presented below.

PERSON	People including fictional
NORP	Nationalities or religious or political groups
FACILITY	Buildings, airports, highways, bridges, etc.
ORGANIZATION	Companies, agencies, institutions, etc.
GPE	Countries, cities, states
LOCATION	Non-GPE locations, mountain ranges, bodies of water
PRODUCT	Vehicles, weapons, foods, etc. (Not services)
EVENT	Named hurricanes, battles, wars, sports events, etc.
WORK OF ART	Titles of books, songs, etc.
LAW	Named documents made into laws
LANGUAGE	Any named language
DATE	Absolute or relative dates or periods
TIME	Times smaller than a day
PERCENT	Percentage (including "%")
MONEY	Monetary values, including unit
QUANTITY	Measurements, as of weight or distance
ORDINAL	"first", "second"
CARDINAL	Numerals that do not fall under another type

Figure 1: All Tags the NER models are trained on. More Info found [here](#).

➤ Tag usage:

Deep Pavlov NER-models use a combination of “B-“ and “I-“Tags to signal what role a word has in a group of tags. If a word does not fit any of the tags in Figure 1, it will be tagged as “O”. Here is an example from the official Deep Pavlov Documentation.

Furthermore, to distinguish adjacent entities with the same tag many applications use BIO tagging scheme. Here “B” denotes beginning of an entity, “I” stands for “inside” and is used for all words comprising the entity except the first one, and “O” means the absence of entity. Example with dropped punctuation:

Bernhard	B-PER
Riemann	I-PER
Carl	B-PER
Friedrich	I-PER
Gauss	I-PER
and	O
Leonhard	B-PER
Euler	I-PER

In the example above PER means person tag, and “B-” and “I-” are prefixes identifying beginnings and continuations of the entities. Without such prefixes, it is impossible to separate Bernhard Riemann from Carl Friedrich Gauss.

Figure 2: example for possible Tags found [here](#).

❖ Installation Guide

➤ Install Anaconda:

1. Go to <https://www.anaconda.com/products/individual> and download the version for your operating system.



Individual Edition

Your data science toolkit

With over 25 million users worldwide, the open-source Individual Edition (Distribution) is the easiest way to perform Python/R data science and machine learning on a single machine. Developed for solo practitioners, it is the toolkit that equips you to work with thousands of open-source packages and libraries.

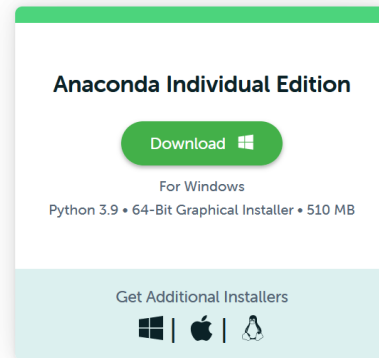


Figure 3: Download button for Anaconda Individual Edition on the official anaconda website

2. Install the downloaded packages at the preferred location. Standard location is: `C:/Users/{Username}/anaconda3`

➤ Install Deep Pavlov:

1. After installing Anaconda open the “Anaconda Prompt” either by typing in “Anaconda prompt” in your search bar or by opening the “Anaconda Navigator” and opening the “CMD.exe Prompt”.

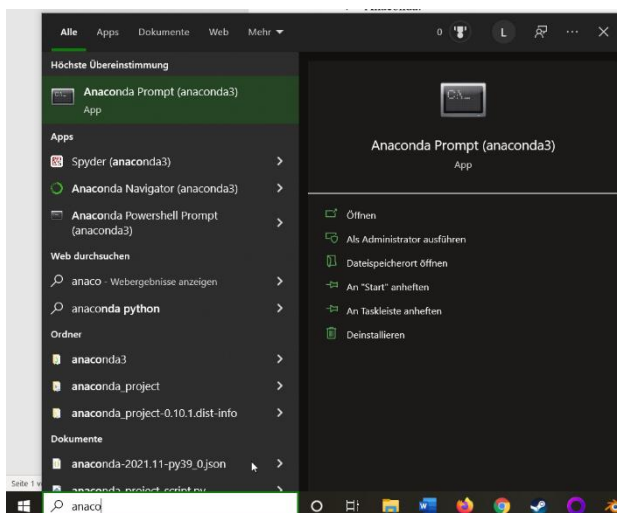


Figure 5: opening the „Anaconda Prompt” with the windows search bar

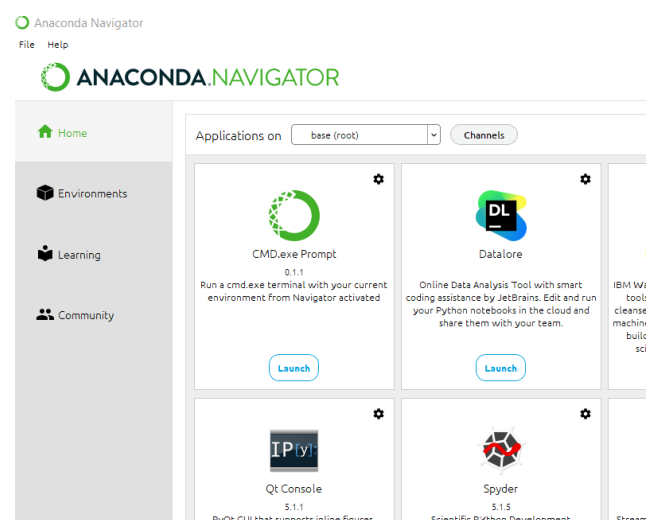
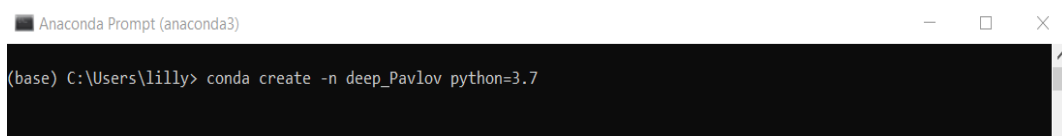


Figure 4: opening the Prompt by using the Anaconda Navigator

2. **Install a new Environment** for Deep Pavlov from the Anaconda Prompt by entering the command “conda create -n {my_env_name} python = 3.7” where {my_env_name} is the name of your environment. For example, “deep_Pavlov”. Then press enter for the command to act.



```

Anaconda Prompt (anaconda3)
(base) C:\Users\lilly> conda create -n deep_Pavlov python=3.7

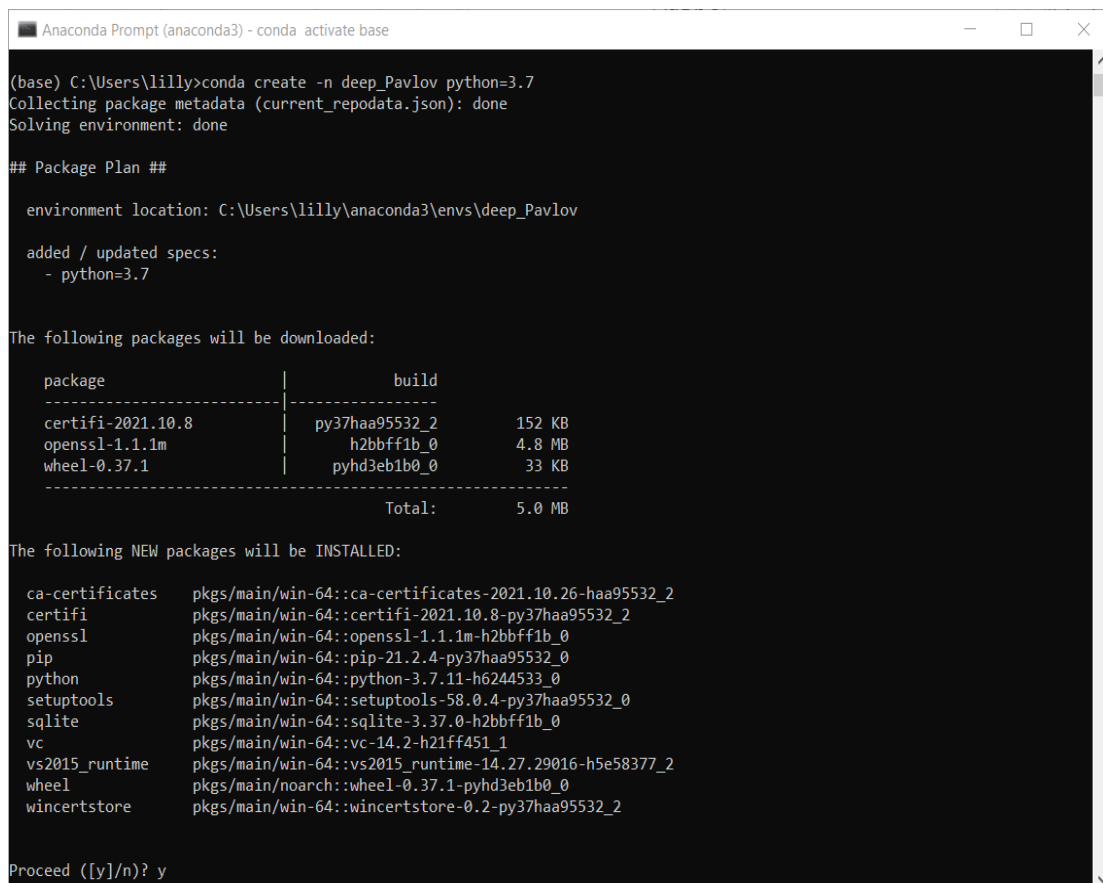
```

Figure 6: Anaconda Prompt with command to create a new environment named “deep_Pavlov” that uses python version 3.7

Tipp
: If
you

want to know more about installation parameters, you can always check out the official conda-Documentation here: <https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#creating-an-environment-with-commands>

You will be asked to agree to the installation of needed packages. When the line “Proceed([y]/n)” appears, type in “y” for yes or “n” for no.



```

Anaconda Prompt (anaconda3) - conda activate base
(base) C:\Users\lilly>conda create -n deep_Pavlov python=3.7
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\lilly\anaconda3\envs\deep_Pavlov

added / updated specs:
- python=3.7

The following packages will be downloaded:

package                        | build                | size
-----|-----|-----
certifi-2021.10.8              | py37haa95532_2      | 152 KB
openssl-1.1.1m                 | h2bbff1b_0          | 4.8 MB
wheel-0.37.1                   | pyhd3eb1b0_0        | 33 KB
-----|-----|-----
Total:                          |                      | 5.0 MB

The following NEW packages will be INSTALLED:

ca-certificates                pkgs/main/win-64::ca-certificates-2021.10.26-haa95532_2
certifi                       pkgs/main/win-64::certifi-2021.10.8-py37haa95532_2
openssl                       pkgs/main/win-64::openssl-1.1.1m-h2bbff1b_0
pip                           pkgs/main/win-64::pip-21.2.4-py37haa95532_0
python                        pkgs/main/win-64::python-3.7.11-h6244533_0
setuptools                    pkgs/main/win-64::setuptools-58.0.4-py37haa95532_0
sqlite                        pkgs/main/win-64::sqlite-3.37.0-h2bbff1b_0
vc                             pkgs/main/win-64::vc-14.2-h21ff451_1
vs2015_runtime                pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
wheel                         pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0
wincertstore                  pkgs/main/win-64::wincertstore-0.2-py37haa95532_2

Proceed ([y]/n)? y

```

Figure 7: Installing necessary packages. Type in “y” if you agree.

3. **Activate the environment** you have just created by entering the command “conda activate {my_env_name}”. where {my_env_name} is the name of your environment. For example, “deep_Pavlov”

```
(base) C:\Users\lilly>conda activate deep_Pavlov
(deep_Pavlov) C:\Users\lilly>
```

Figure 8: Command to activate an environment with the name “deep_Pavlov”.

Tip: the name in brackets on the left - for example “(base)” or “(deep_Pavlov)” - tells you which environment you are currently in.

Tip: If you ever forget the name of your environment, enter the command “conda info --envs” and you will get the name and location of all installed environments.

```
(base) C:\Users\lilly>conda info --envs
# conda environments:
#
base                *  C:\Users\lilly\anaconda3
Deep_P_env          C:\Users\lilly\anaconda3\envs\Deep_P_env
deep_Pavlov         C:\Users\lilly\anaconda3\envs\deep_Pavlov
```

Figure 9: In this example, there are three environments installed. The “base”-environment, an environment called “Deep_P_env” and the environment called “deep_Pavlov” we just installed.

4. **Install**

DeepPavlov by entering the command “pip install deeppavlov” from within your active environment.

```
(deep_Pavlov) C:\Users\lilly>pip install deeppavlov
```

Figure 10: Installing deeppavlov to your environment. Make sure to be in the correct environment! For example (deep_Pavlov).

Congratulations! You have now installed Deep Pavlov!

➤ Install NER – Model

1. After successfully installing Deep Pavlov, you can now install the Models you want to use. For our Datasets we used the model “ner_ontonotes_bert_mult” which manages NER-Tagging for multiple languages. You can find an overview of available NER-Models at: <http://docs.deeppavlov.ai/en/master/features/models/ner.html>
2. To install a model, use the command “python -m deeppavlov install {config_path}” from within your environment where {config_path} is the name of the chosen model’s configuration file. You can find the names of the config files [here](#) under “list of all available configs”.
Example: To install our model, we used the command “python -m deeppavlov install ner_ontonotes_bert_mult”.

➤ Download Data for the pretrained model

1. For the model to run smoothly without further training, best activate the model once by using the command “python -m deeppavlov interact ner_ontonotes_bert_mult -d” from within your environment. This will download all required data in case of updates etc. Once “x::” appears, the model is ready for processing input. For now, type “exit” and press enter to exit the model and then close the “Anaconda Prompt” window.

```
WARNING:tensorflow:From C:\Users\lilly\anaconda3\envs\deep_P_env\lib\site-packages\deeppavlov\models\bert\bert_sequence_tagger.py:249: checkpoint_exists (from tensorflow.python.training.checkpoint_management) is deprecated and will be removed in a future version.
Instructions for updating:
Use standard file APIs to check for files with this prefix.
2022-01-27 16:14:08.889 INFO in 'deeppavlov.core.models.tf_model'['tf_model'] at line 51: [loading model from C:\Users\lilly\deeppavlov\models\ner_ontonotes_bert_mult\model]
WARNING:tensorflow:From C:\Users\lilly\anaconda3\envs\deep_P_env\lib\site-packages\deeppavlov\core\models\tf_model.py:54: The name tf.train.Saver is deprecated. Please use tf.compat.v1.train.Saver instead.
x::
```

Figure 11: NER-Model activated and ready for input

Tip: this step may take a while. Don't worry.

➤ Usage with Script

▪ Before you run the model for the first time

Copy the vocab.txt to the project folder.

The vocab.txt is a file containing tagging information that the python script needs to generate output. Copy the file to the project folder.

The file is found at:
C:\Users\{username}\.deeppavlov\downloads\bert_models\multi_cased_L-12_H-768_A-12\vocab.txt.

The project folder is the folder containing the DP_Dostojewski.py and input.txt files, and where the output-files will be saved in.

▪ Running the Model

- Open the “Anaconda Prompt” as described [above](#) and activate your environment as described [above](#) with the command “conda activate {my_env_name}”
- Navigate to where the Python File is located (if necessary)
- Run the python File with the command “python {name_of_file.py}” for example, “DP_Dostojewski.py”.

➤ File System

▪ Input

The input file is read by the model. Copy your text into the “input.txt” file and save it. The next time you use the model by running the provided python script, the new text will be used as input.

▪ Output

The output files will be in the same folder as the Python script and the input file. The names stand for the following:

- output_merged.txt: the full input text tagged and comma-separated word by word
- output_geo.txt: all words that have geo-tags, with their tag.
- output_geo_context.txt: all words that have starting geo-tags, with their tag and the context they were found in (some words before and after the word)

Reminder: an explanation of the tags is found in Figure 1 and Figure 2.

Tip: while testing our script, it would run an analysis of a text of 5 pages in under a minute, while the complete text of Crime and Punishment took around 1 hour to complete. Best test the model & script with a small text first and only then go on to a bigger corpus.

❖ Limitations & possible Additions

The appendant script separates the sentences to fit the token limit of the model. It may happen that named Entities are separated by this. A solution could be to separate the text into sentences instead. Then you could still encounter sentences that exceed the limit of 512 tokens, but the probability for named Entities cut in half would be lower. We tried to write such an alternative script, but alas, the model took a big amount longer to run and encountered various errors, we did not manage to fix for the time being. Therefore, we continued with the version separating the text by token limit. A script separating both by sentences and token limit would be a valuable continuation and addition to this project.

So far there is no way implemented to refer to the context of a found Named Entity, namely a page, chapter, or line. Context is provided in the `output_geo_context.txt` file, but only by adding preceding and following words.

❖ Alternatives to Deep Pavlov

Possible alternatives to Deep Pavlov are Frameworks like [spaCy](#), [NLTK](#) – “Natural Language Toolkit” and [Flair](#) – State-of-the-Art Natural Language Processing.

When deciding which NER framework to use, it is often useful to consult comparative studies where the frameworks are tested on the same data sets. The results can then be compared with each other to see how well each framework performed in recognizing the entities. The higher

Software	Entity	CoNLL 2003			GMB		
		P	R	F1	P	R	F1
Stanford NLP	LOC	91.30	88.73	90	83.10	63.64	72.08
	ORG	86.32	80.92	83.53	71.40	47.42	56.99
	PER	92.72	82.68	87.41	78.59	84.70	81.53
	Overall	90.06	73.67	81.05	79.81	63.74	70.88
NLTK	LOC	52.47	65.47	58.26	77.13	77.1	77.12
	ORG	36.20	24.80	29.44	42.06	35.54	38.53
	PER	61.09	66.11	63.50	38.07	55.87	45.28
	Overall	51.78	45.56	48.47	60.96	63.91	62.40
Gate	LOC	59.63	78.63	67.82	79.03	48.16	59.85
	ORG	50.58	21.29	29.96	45.08	37.68	41.05
	PER	69.53	62.67	65.92	46.53	53.70	49.86
	Overall	61.48	47.44	53.55	61.72	46.78	53.22
OpenNLP	LOC	76.54	52.22	62.08	84.34	45.84	59.40
	ORG	38.06	14.87	21.39	59.27	30.64	40.39
	PER	83.94	37.17	51.52	62.34	41.98	50.17
	Overall	68.68	30.44	42.18	37.35	41.71	39.41
SpaCy	LOC	73.38	75.36	74.36	77.04	56.64	65.28
	ORG	40.95	36.24	38.45	41.20	36.50	38.70
	PER	66.89	56.22	61.09	67.41	69.14	68.27
	Overall	60.94	49.01	54.33	66.15	54.32	59.66

Figure 13: F1 score of multiple NER Frameworks. Source: Schmitt et al. (2019)

the F1 score, the better the NER framework works.

Deep Pavlov F1

TOTAL	79.39
PER	95.74
LOC	82.62
ORG	55.68

Figure 14: Deep Pavlovs F1 score for the Person, Location and Organization Tag

❖ Links for further Information

Deep Pavlov Documentation:

<https://docs.deeppavlov.ai/en/master/index.html>

Installation guide:

<http://docs.deeppavlov.ai/en/master/intro/installation.html>

Modell overview:

<http://docs.deeppavlov.ai/en/master/features/models/ner.html>

Information for NER-Models:

<https://docs.deeppavlov.ai/en/master/features/models/ner.html#>

F1 scores Deep Pavlov:

<https://docs.deeppavlov.ai/en/master/features/models/ner.html#multilingual-bert-zero-shot-transfer>

Github config for ner_ontonotes_bert_mult modell:
https://github.com/deepmipt/DeepPavlov/blob/0.17.2/deeppavlov/configs/ner/ner_ontonotes_bert_mult.json

Demo of NER:

<https://demo.deeppavlov.ai/#/en/ner>

Official Conda Documentation:

<https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html#creating-an-environment-with-commands>

F1 score of NER Frameworks:

Schmitt x. et al., (2019). *A Replicable Comparison Study of NER Software: StanfordNLP, NLTK, OpenNLP, SpaCy, Gate*. 2019. Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS).

from: https://www.researchgate.net/profile/Sylvain-Kubler/publication/337977695_A_Replicable_Comparison_Study_of_NER_Software_StanfordNLP_NLTK_OpenNLP_SpaCy_Gate/links/5f52b5d3299bf13a31a07658/A-Replicable-Comparison-Study-of-NER-Software-StanfordNLP-NLTK-OpenNLP-SpaCy-Gate.pdf,
visited at 30.01.2022

Project Github Repository:

https://github.com/lruehle/NER_DeepPavlov

Interesting Project about tagging Russian Dostoevsky tags:

<https://digitaldostoevsky.com/2021/08/05/encoding-dostoevsky/>