Learn Git and GitHub without any code!

Using the Hello World guide, you'll start a branch, write comments, and open a pull request.

Read the guide

inspirehep / magpie

Deep neural network framework for multi-label text classification

#neural-network #nlp #deep-learning #word2vec #classification #multi-label-classification #machine-learning #prediction

196 commits | 5 branches | 0 packages | 4 releases | 5 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

jstypka Update README.md — Latest commit 29dac81 on 8 Apr

data	api: add the test_split parameter and use it in the IT	3 years ago
docs/img	Update the logo	4 years ago
magpie	Bump deps	10 months ago

.gitignore	ontology: rewrite the class, massive speedup, add dependencies	5 years ago
.travis.yml	Bump deps	10 months ago
LICENSE	Create LICENSE	5 years ago
README.md	Update README.md	2 months ago
setup.py	Bump dependencies (#195)	2 months ago

README.md

Magpie is a deep learning tool for multi-label text classification. It learns on the training corpus to assign labels to arbitrary text and can be used to predict those labels on unknown data. It has been developed at CERN to assign subject categories to High Energy Physics abstracts and extract keywords from them.

Very short introduction

```
>>> magpie = Magpie()
>>> magpie.init_word_vectors('/path/to/corpus', vec_dim=100)
>>> magpie.train('/path/to/corpus', ['label1', 'label2', 'label3'], epochs=3)
Training...
>>> magpie.predict_from_text('Well, that was quick!')
[('label1', 0.96), ('label3', 0.65), ('label2', 0.21)]
```

Short introduction

To train the model you need to have a large corpus of labeled data in a text format encoded as UTF-8. An example corpus can be found under data/hep-categories directory. Magpie looks for `.txt` files containing the text to predict on and corresponding `.lab` files with assigned labels in separate lines. A pair of files containing the labels and the text should have the same name and differ only in extensions e.g.

```
$ ls data/hep-categories
1000222.lab 1000222.txt 1000362.lab 1000362.txt 1001810.lab 1001810.txt ...
```

Before you train the model, you need to build appropriate word vector representations for your corpus. In theory, you can train them on a different corpus or reuse already trained ones (tutorial), however Magpie enables you to do that as well.

```
from magpie import Magpie
```

```
magpie = Magpie()
magpie.train_word2vec('data/hep-categories', vec_dim=100)
```

Then you need to fit a scaling matrix to normalize input data, it is specific to the trained word2vec representation. Here's the one liner:

```
magpie.fit_scaler('data/hep-categories')
```

You would usually want to combine those two steps, by simply running:

```
magpie.init_word_vectors('data/hep-categories', vec_dim=100)
```

If you plan to reuse the trained word representations, you might want to save them and pass in the constructor to `Magpie` next time. For the training, just type:

```
labels = ['Gravitation and Cosmology', 'Experiment-HEP', 'Theory-HEP']
magpie.train('data/hep-categories', labels, test_ratio=0.2, epochs=30)
```

By providing the `test_ratio` argument, the model splits data into train & test datasets (in this example into 80/20 ratio) and evaluates itself after every epoch displaying it's current loss and accuracy. The default value of `test_ratio` is 0 meaning that all the data will be used for training.

If your data doesn't fit into memory, you can also run `magpie.batch_train()` which has a similar API, but is more memory efficient.

Trained models can be used for prediction with methods:

```
>>> magpie.predict_from_file('data/hep-categories/1002413.txt')
[('Experiment-HEP', 0.47593361),
```

```
 ('Gravitation and Cosmology', 0.055745006),
 ('Theory-HEP', 0.02692855)]

>>> magpie.predict_from_text('Stephen Hawking studies black holes')
[('Gravitation and Cosmology', 0.96627593),
 ('Experiment-HEP', 0.64958507),
 ('Theory-HEP', 0.20917746)]
```

Saving & loading the model

A `Magpie` object consists of three components - the word2vec mappings, a scaler and a `keras` model. In order to train Magpie you can either provide the word2vec mappings and a scaler in advance or let the program compute them for you on the training data. Usually you would want to train them yourself on a full dataset and reuse them afterwards. You can use the provided functions for that purpose:

```
magpie.save_word2vec_model('/save/my/embeddings/here')
magpie.save_scaler('/save/my/scaler/here', overwrite=True)
magpie.save_model('/save/my/model/here.h5')
```

When you want to reinitialize your trained model, you can run:

```
magpie = Magpie(
    keras_model='/save/my/model/here.h5',
    word2vec_model='/save/my/embeddings/here',
    scaler='/save/my/scaler/here',
    labels=['cat', 'dog', 'cow']
)
```

or just pass the objects directly!

Installation

The package is not on PyPi, but you can get it directly from GitHub:

```
$ pip install git+https://github.com/inspirehep/magpie.git@v2.1.1
```

If you encounter any problems with the installation, make sure to install the correct versions of dependencies listed in `setup.py` file.

Disclaimer & citation

The neural network models used within Magpie are based on work done by Yoon Kim and subsequently Mark Berger.

Contact

If you have any problems, feel free to open an issue. We'll do our best to help 👍