Joseph Birkner
Mat.Nr. 11393024

Protokoll zum Repetitorium (22.05.2017)

Dozent: Benjamin Roth

„Datenvisualisierung mit Pyplot“

Durch Frameworks wie Scikit, Numpy, Gensim, NLTK, Theano und Tensor Flow hat sich Python zu einer etablierten Platform zur Implementierung computerlinguistischer Modelle entwickelt. Unter diesem Gesichtspunkt ist es wichtig zu wissen, wie man Modelle und Ergebnisse zügig und ansprechend mit Python visualisieren kann.

Als ein sehr nützliches Python-Modul zur Erfüllung dieser Anforderung wurde *matplotlib* vorgestellt, welches die Bibliothek *pyplot* enthält. *pyplot* verwendet, ähnlich wie auch viele der oben genannten Modell-Frameworks Datenstrukturen aus numpy, was die Einbindung also stark vereinfacht.

Graphen in pyplot werden stateful erzeugt. Das bedeutet, die einzelnen Methoden der Library manipulieren eine interne Repräsentation, die später mit pyplot.show() oder pyplot.savefig() „abgeholt“ werden kann. Ein Graph kann mittels der plotly.plot(x_array, y_array, [color, linewidth, linestyle]) in das Koordinatensystem eingefügt werden.

Weitere nützliche Methoden:

pyplot.figure()	Setzen von Größe und DPI der auszugebenden Visualisierung.
pyplot.xlim(), pyplot.ylim()	Setzen der Darstellungsausschnitte der jeweiligen Achsen.
pyplot.xticks(), pyplot.yticks()	Setzen von Achsenbeschriftungen für einzelne Koordinaten.
pyplot.legend()	Konfiguration der anzuzeigenden Legende.
pyplot.annotate()	Beschriftung eines einzelnen Punktes im Graphen, möglw. mit Latex.

Weiterhin wurden auch Scatter (Streu-) Diagramme vorgestellt, wobei i.d.F. zum Einfügen der Daten einfach die pyplot.scatter() Methode verwendet werden kann. In der Compterlinguistik werden solche Diagramme häufig zur Darstellung hochdimensionaler Objekte (Wort/Dokumentvektoren) verwendet. Um diese Hochdimensionalen Daten sinnvoll in zwei Dimensionen konvertieren zu können, wurde außerdem der t-SNE Algorithmus genannt, für den das Scikit-Learn Modul eine Implementierung bietet.

Abschließend wurden außerdem die Plotarten pyplot.hist() (Nützlich für Frequenzverteilungen) und pyplot.bar() (Nützlich zur vergleichenden Darstellung von Accuracy/Precision/Recall verschiedener Modelle) vorgestellt.