SeemsPhishy

NLU-Evaluation

In der folgenden Dokumentation werden die in SeemsPhishy verwendeten Modelle zum Natural Language Understanding (NLU) evaluiert und teilweise mit anderen verglichen. Allgemeines Ziel war es Schlagwörter für die Text-Generation zu erzeugen.

Keywords

Zur Ermittlung der Keywords wurden zwei Ansätze verglichen. Für die Ansätze wurden Yake und Rake-Nltk verwendet. Um die Ansätze zu vergleichen werden die Modelle auf demselben Input getestet und der Output miteinander, und mit den erwarteten Keywords verglichen.

Als Input wird folgender Beispiel-Text verwendet:

„spaCy is an open-source software library for advanced natural language processing, written in the programming languages Python and Cython. The library is published under the MIT license and its main developers are Matthew Honnibal and Ines Montani, the founders of the software company Explosion.“

Folgende Keywords wären zu erwarten: **[Spacy, Python, Cython, open-source, natural language processing, Matthew Honnibal, Ines Montani, software company Explosion]**

- Der Output von **Rake-Nltk**:
  - ['advanced natural language processing', 'software company explosion', 'programmin g languages python', 'source software library', 'mit license', 'matthew honnibal', 'mai n developers', 'ines montani', 'library', 'written', 'spacy', 'published', 'open', 'founders ', 'cython']
  - **6/9 Keywords erkannt**

- Der Output von **Yake** hierzu ist:
    - ('programming languages Python', 0.001295347548560416)
    - ('natural language processing', 0.002012136772192602)
    - ('advanced natural language', 0.0026621455770583914)
    - ('Python and Cython', 0.0035840985079775055)
    - ('open-source software library', 0.008298152696966859)
    - ('languages Python', 0.009390717577572831)
    - ('language processing', 0.01453240965208459)
    - ('software company Explosion', 0.015993140254256993)
    - ('advanced natural', 0.01840251352140607)
    - ('natural language', 0.019161829017826378)
    - ('programming languages', 0.019161829017826378)
    - ('open-source software', 0.032652195076937375)
    - ('Ines Montani', 0.03375876229391358)
    - ('Matthew Honnibal', 0.04096703831447956)
    - ('Honnibal and Ines', 0.04096703831447956)
    - ('Cython', 0.053691021027863564)
    - ('software library', 0.05857047036380304)
    - ('company Explosion', 0.06120870235178475)
    - ('Python', 0.06651575167590484)
    - ('library for advanced', 0.07441175006256819)
    - **9/9 Keywords erkannt**

Beide Modelle schneiden gut ab. Allerdings liegt Yake doch klar erkennbar vorne. Um die Modelle weiter zu vergleichen, nehmen wir einen weiteren nah am Anwendungsbeispiel liegenden Input aus einem Text von Apple:

„The Apple identity is a seal of approval and a promise of excellence. When you are authorized or certified in your area of business or expertise, you also represent Apple. By following these guidelines, you reap the benefits of the Apple identity and contribute to its strength.“

Erwartete Keywords: **[Apple, seal of approval, promise of excellence, guidelines, Apple identity]**

- Der Output von **Rake-Nltk**:
    - ['also represent apple', 'apple identity', 'apple identity', 'strength', 'seal', 'reap', 'promise', 'guidelines', 'following', 'expertise', 'excellence', 'contribute', 'certified', 'business', 'benefits', 'authorized', 'area', 'approval']
    - **2/5 Keywords erkannt**

- Der Output von **Yake** hierzu:
    - ('promise of excellence', 0.02617581190902736)
    - ('Apple identity', 0.032484786288018264)
    - ('seal of approval', 0.03588587835699325)
    - ('Apple', 0.057956023860663765)
    - ('represent Apple', 0.08338318833688906)
    - ('excellence', 0.13704795315808577)
    - ('identity', 0.17543772098990723)
    - ('seal', 0.18612548139510782)

- ('approval', 0.18612548139510782)
- ('promise', 0.18612548139510782)
- ('business or expertise', 0.18675169110600065)
- ('authorized or certified', 0.24370924106100242)
- ('area of business', 0.24370924106100242)
- ('expertise', 0.3554905954934143)
- ('reap the benefits', 0.4040201998862045)
- ('identity and contribute', 0.42691931061881777)
- ('authorized', 0.4426664094849176)
- ('certified', 0.4426664094849176)
- ('area', 0.4426664094849176)
- ('business', 0.4426664094849176)

- **4/5 Keywords wurden erkannt**

Beide verwendeten Modelle eignen sich für den Einsatz, allerdings trifft Yake in beiden Szenarios mehr Keywords. Ein weiterer Vorteil sind die Werte, wie wichtig Yake die jeweiligen Keywords findet und die Möglichkeit die Größe der Keywords, sowie die Dopplungen innerhalb der Keywords zu verringern. Damit ließe sich in Zukunft eine vollautomatisierte Text-Generierung realisieren.

TF-IDF

Im Bereich TF-IDF verursachte weniger die Auswahl des besseren Modells, sondern vielmehr die korrekte Anbindung an das System, bzw. der Aufwand Probleme. Der Vectorizer von Sklearn konnte mit lediglich 8 Zeilen Code gleich auf den ersten Blick überzeugen. Ein anderes Modell wie zum Beispiel TF-IDF mit Spacy sprengte mit ca. 300 Zeilen deutlich den Rahmen und die zeitlichen Möglichkeiten.

Erste Versuche mit Vectorizer von Sklearn über mehrere Texte von Apple brachten Folgenden Output:

['product', 'apple', 'inc', 'customer', 'device', 'establish', 'computer', 'provide', 'company', 'cellphone', ' television', 'order', 'executive', 'creation', 'quality', 'transition', 'range', 'assurance', 'retail', 'seek', 'cha in', 'summary', 'wearable', 'well', 'wide']

Tatsächlich sind die Keywords für sich genommen teilweise eher wenig sagend. Allerdings bieten sie eine gute Ergänzung zu den sehr zentralen Schlagwörtern von Yake. Als Beispiel können inhaltlich zu den Keywords von Yake „Apple identity“, „promise of excellence“ und „seal of approval“ die von tf-idf gefundenen Wörter „product“ oder „computer“ gematched werden, um den einzelnen Schlagwörtern mehr Zusammenhang zu vermitteln. Aufgrund dessen wird tf-idf mit sklearn in SeemsPhishy eingesetzt.

Named Entity Recognition

Named Entity Recognition (NER) ist in der Regel in vielen NLU Prozessen ein Bestandteil. Für den Use-Case in SeemsPhishy ist NER allerdings schwierig einzusetzen. Ein Hauptproblem hierbei ist, dass mehrere große Dokumente mit jeweils sehr vielen eigen Namen analysiert werden.

Ein Beispiel hierzu: Durchlauf der NER von Spacy mit 20 Dokumenten hatte ca. 13.000 Entities als Ergebnis. Somit deutlich zu viele und auch teilweise völlig unbedeutende Entities.

Für den Benutzer wäre eine der Art große Zahl nicht sinnvoll einzusetzen. Außerdem kann die Text-Generation ohnehin nur schwierig mit speziellen Eigennamen arbeiten, da sie diese oft noch nicht kennt. Dazu kommt, dass Yake bereits die wichtigsten Entities identifiziert. Somit empfehlen wir auf eine NER auf der SeemsPhishy Webseite zu verzichten, weswegen NER eine untergeordnete Rolle spielt.

Dennoch wurde die NER mit Spacy an die Webseite angebunden und ist funktional. Eine Verbesserung in diesem Bereich ließe sich durch die NER mit Bert realisieren. Bert lieferte hierbei bei ersten Tests vielversprechendere Ergebnisse und kann außerdem Werte liefern, mit denen sich die Relevanz der unterschiedlichen Entities bestimmen lässt.

Quellen

Vergleiche Notebook: „Evaluation.ipynb“