Linking 10-K and the GICS

through Experiments of Text Classification and Clustering

Tingyue Gan

Consortium for Data Analytics in Risk
University of California, Berkeley

tgan@berkeley.edu

April 16, 2019

Overview

1. Form 10-K and the GICS

2. 10-K Classification with GICS Sectors as Labels
   - A brief introduction to text classification using Convolutional Neural Network (CNN)
   - Fix the text CNN architecture, vary word representations
   - Extract 10-K embeddings

3. Clustering 10-K
   - A brief introduction to the Louvian method for community detection
   - Fix the clustering architecture, vary 10-K representations

Form 10-K

- The Securities and Exchange Commission (SEC) requires every publicly traded company in the US to file a 10-K annual report disclosing financial performance.
- A 10-K is typically long and complicated, but it is one of the most comprehensive and most important documents that a public company publishes each year.
- Every 10-K report contains 4 parts and 15 schedules (items). We focus on the textual information in **Item 1. Business** and **Item 1A. Risk Factors** from Part 1, and sentences containing the word "**revenue**".

The Global Industry Classification Standard (GICS)

- The Global Industry Classification Standard (GICS) is an industry taxonomy developed in 1999 by MSCI and S&P Dow Jones Indices.
- It is designed to classify a company according to its principal business activity, and it uses revenues as a key factor in determining a firm's principal business activity.
- The GICS hierarchy begins with 11 sectors and is followed by 24 industry groups, 68 industries, and 157 sub-industries. Our primary focus is the classification at the **11-sector level**.

Convolutional Neural Network (CNN) for text classification

- CNN is a class of deep neural networks, most commonly applied to analyzing visual imagery. (The hidden layers of a CNN typically consist of convolutional layers, RELU layer (activation), pooling layers, fully connected layers and normalization layers.)
- CNN requires minimal preprocessing and automatically learns filters that, in traditional algorithms, were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage. (For example, in Image Classification, a CNN can learn to detect edges from raw pixels in the first layer, then use the edges to detect simple shapes in the second layer, and then use these shapes to detect higher-level features. The last layer is then a classifier that uses these high-level features.)
- CNN for text classification [1]
  1. Represent a word by a vector.
  2. A text document is a sequence of words, we can construct a document matrix by stacking (in order) the word vectors as rows.
  3. The document matrix is then passed as input to the CNN (1D).
     - ⋆ In vision, filters slide over local patches of an image (2D), but in NLP, filters typically slide over full rows of the document matrix (1D).

Convolutional Neural Network (CNN) architecture for text classification

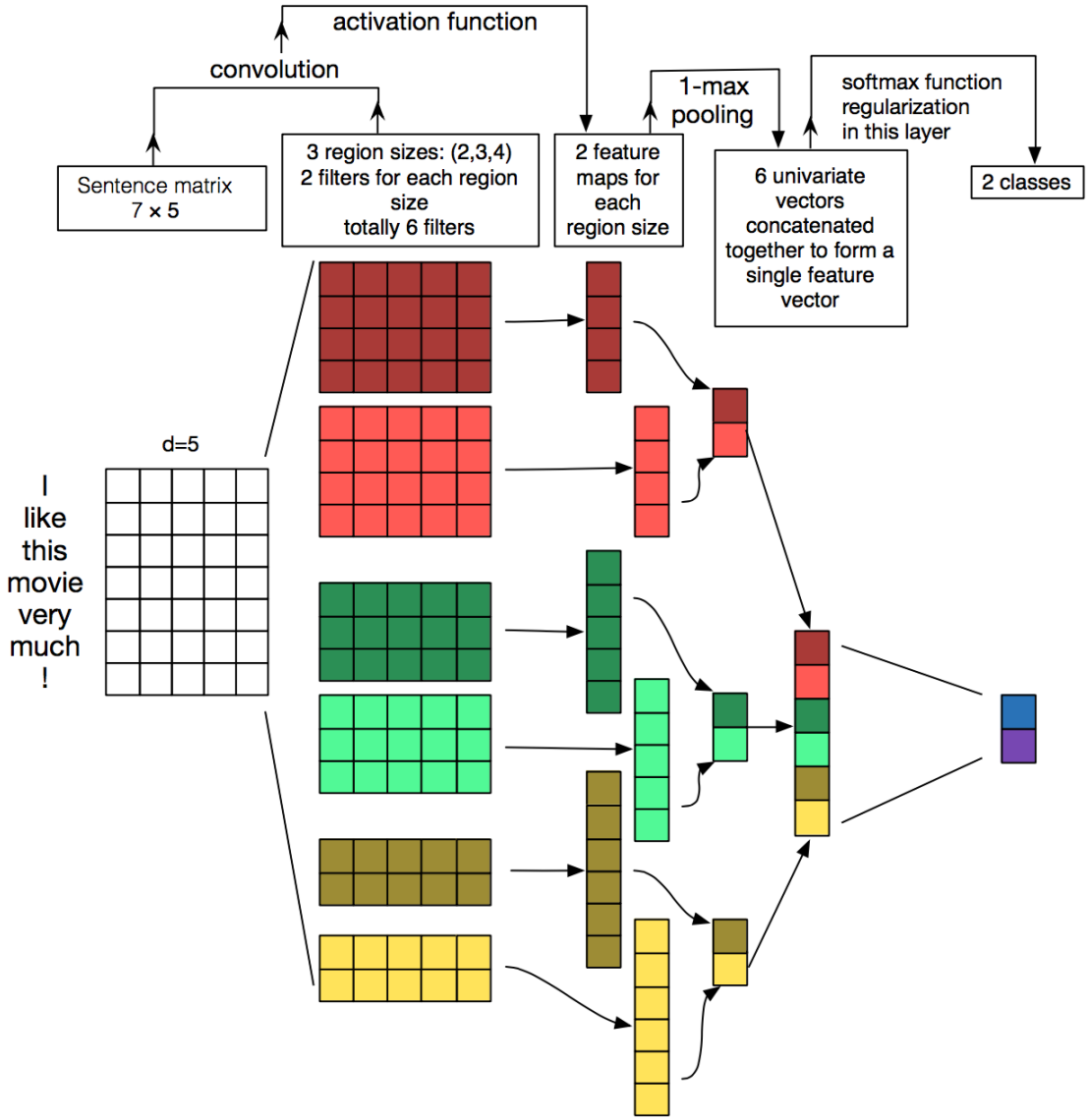


Image Reference : `http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/`

Fix the text CNN architecture, vary word representations

Model	glove300	one300	idf300	randU300
Dimension	300	300	300	300
Vocabulary	400,000	300	300	300
Weightings	pre-trained [2]	$1_{id(w)}$	$1_{id(w)} \log(\frac{1}{df(w)})$	uniform(-1,1)

Table: Four different vector representations for words

How are the 300 words selected in models *one300*, *idf300* and *randU300*?

1. Filter the corpus by word-document-frequency: $df(w) \in [0.05, 0.1]$.
2. Take the 300 words with the highest term-frequency in the filtered corpus.

The text CNN architecture used for training (l3f5)

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 45000)	0
embedding_1 (Embedding)	(None, 45000, 300)	46318800
conv1d_1 (Conv1D)	(None, 44996, 128)	192128
max_pooling1d_1 (MaxPooling1	(None, 8999, 128)	0
conv1d_2 (Conv1D)	(None, 8995, 128)	82048
max_pooling1d_2 (MaxPooling1	(None, 1799, 128)	0
conv1d_3 (Conv1D)	(None, 1795, 128)	82048
global_max_pooling1d_1 (Glob	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dense_2 (Dense)	(None, 11)	1419

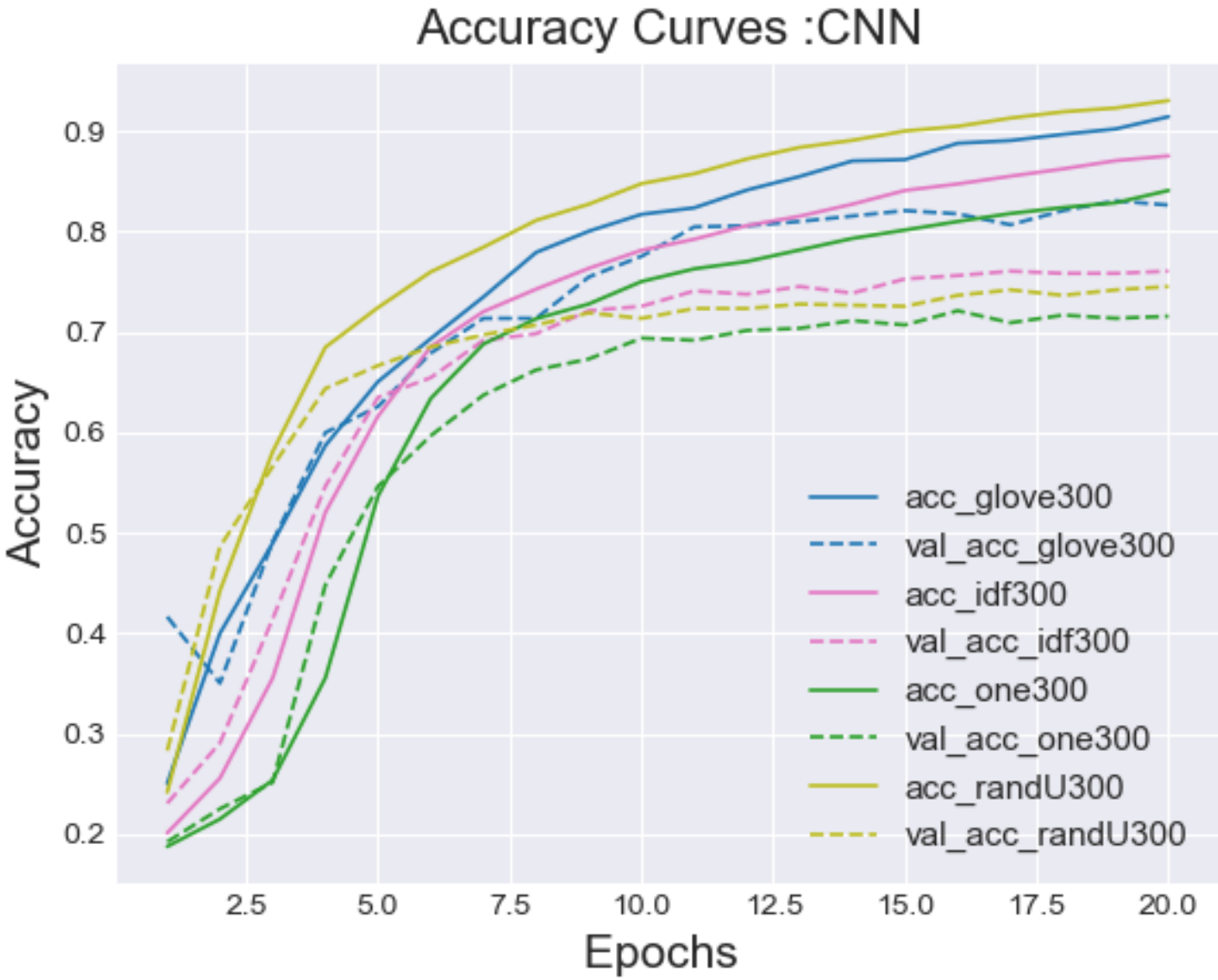
Trainable params: 374,155, Non-trainable params: 46,318,800

2017 10-K dataset (training and validation)

Sectors	Counts
Energy	333
Materials	217
Industrials	558
Consume Discretionary	565
Consumer Staples	187
Health Care	818
Financials	775
Information Technology	645
Telecommunication Services	42
Utilities	193
Real Estate	244

- Total 4577 10-K filings.
- Train-validation split: (0.8, 0.2)
- Primary goal is to extract embeddings of the 2017 10-Ks, but will "test" the best model on 2016 and 2018 10-K datasets.

Training and validation accuracy



Average accuracy

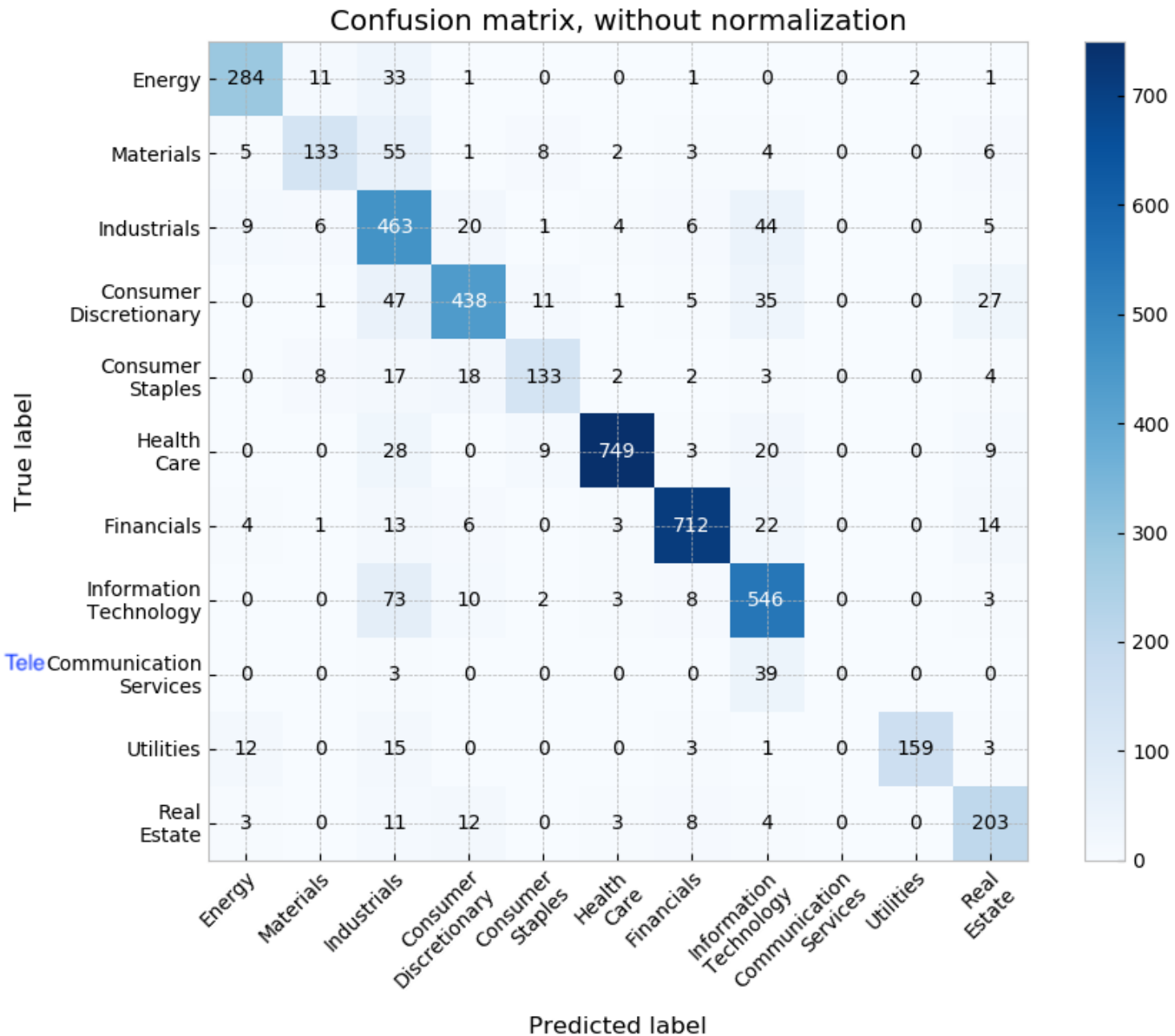
$$\text{average accuracy} = \frac{\text{total number of matches}}{\text{total number of samples}}$$

Model	glove300	one300	idf300	randU300
Training	0.9146	0.8411	0.8755	0.9306
Validation	0.8308	0.7213	0.7607	0.7454

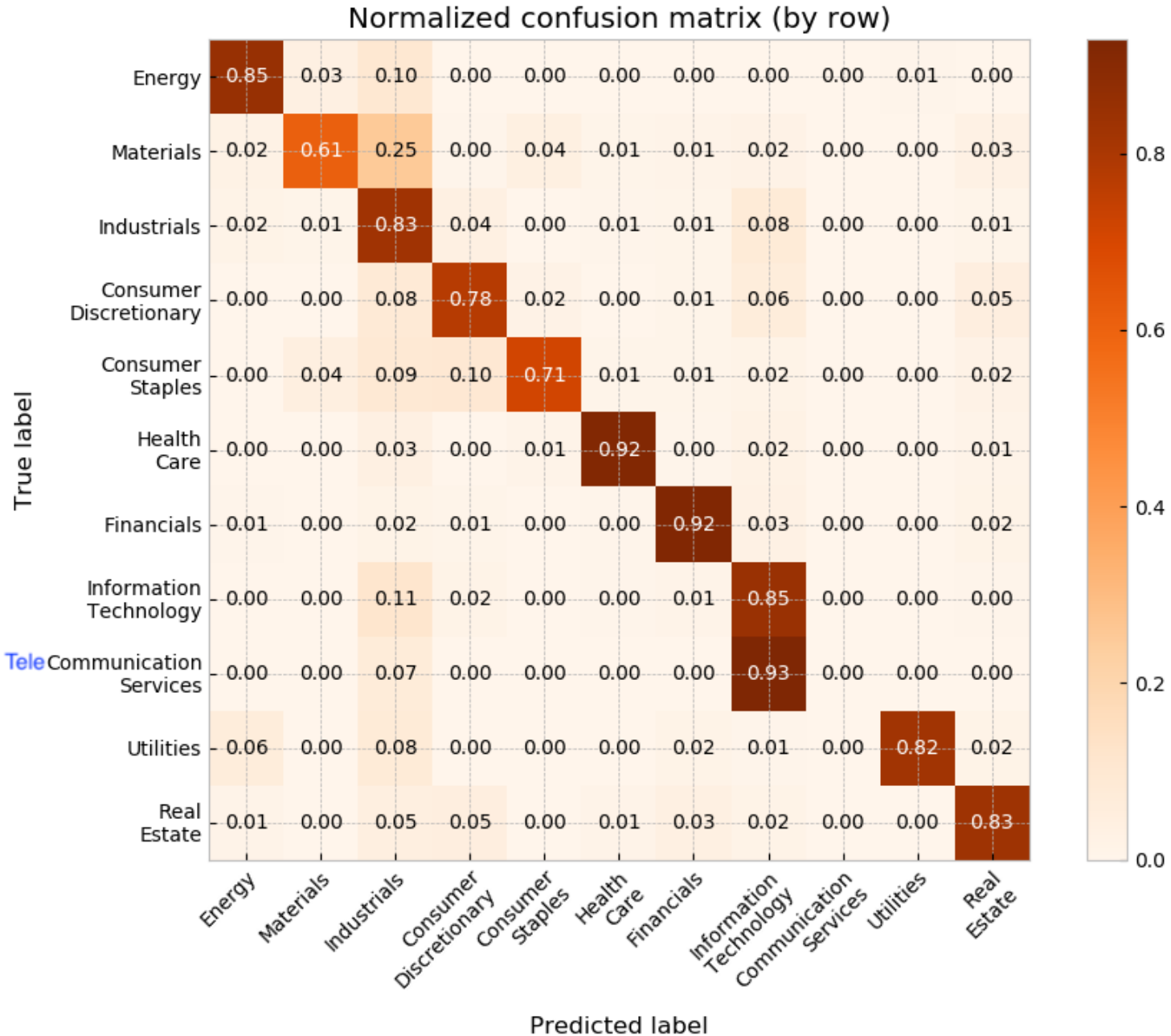
Confusion matrix

- The confusion matrix (CM) shows the ways in which the classification model is confused when it makes predictions.
- The number of correct and incorrect predictions are summarized with count values and broken down by each class:
  $CM(i, j) =$ number of samples with true label i and predicted label j.
- It gives us insight not only into the errors being made by a classifier but more importantly the types of errors that are being made.
- It can be normalized by row or by column to a stochastic matrix

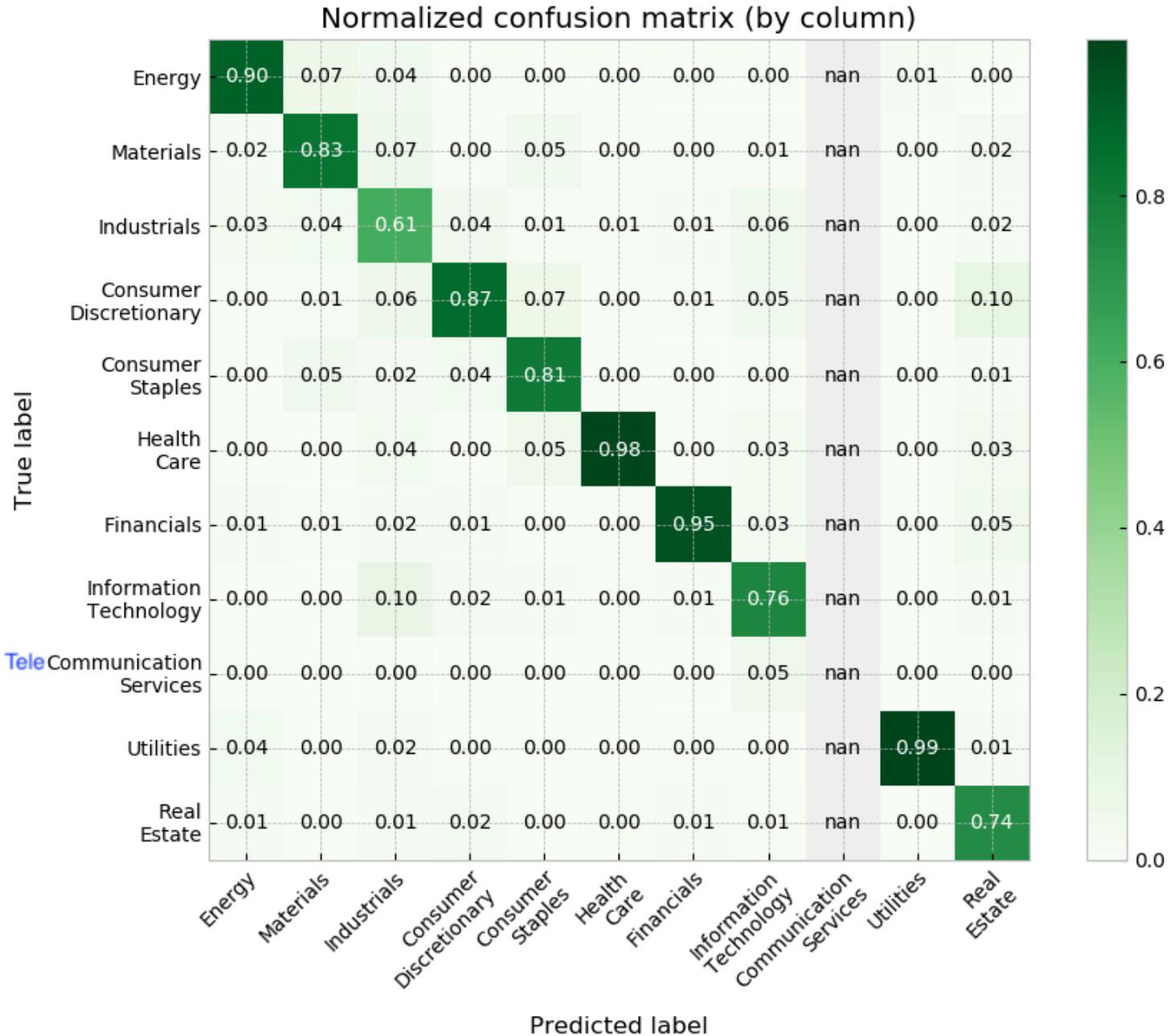
cnn-glove300: Confusion matrix (2017 validation)



cnn-glove300: Confusion matrix normalized by row (2017 validation)



cnn-glove300: Confusion matrix normalized by column (2017 validation)

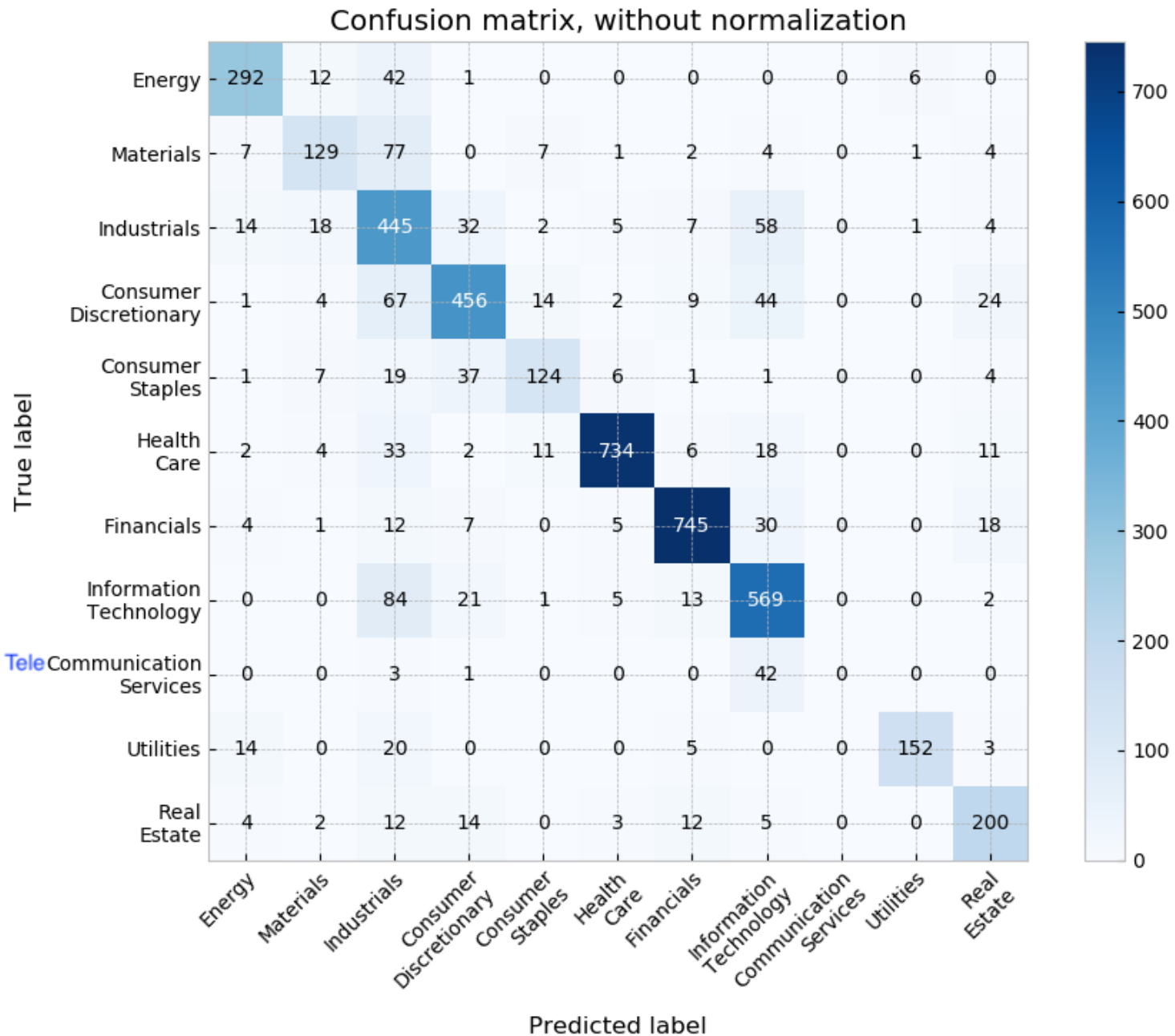


2016 10-K dataset ("test")

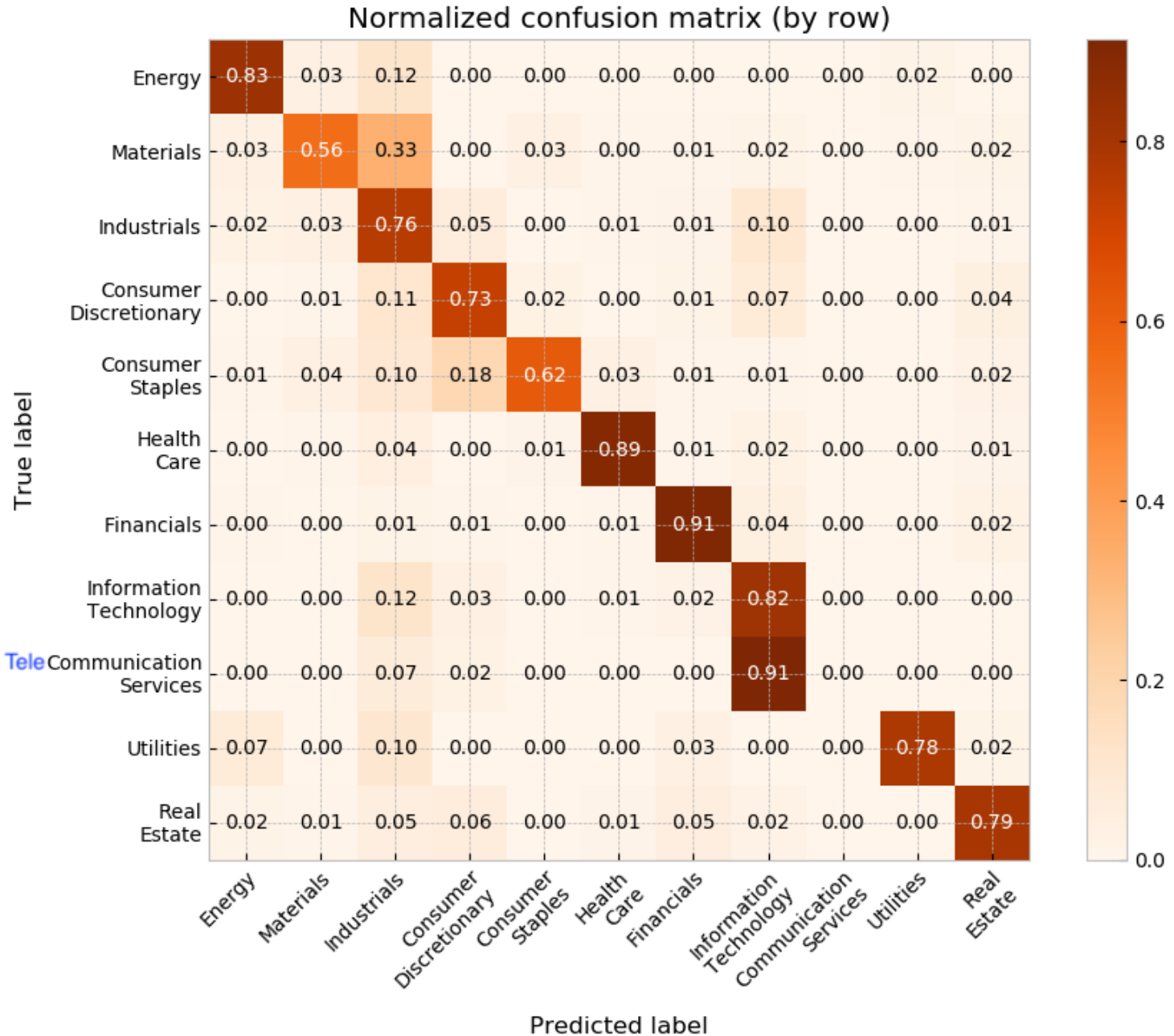
Sectors	Counts
Energy	353
Materials	232
Industrials	586
Consume Discretionary	621
Consumer Staples	200
Health Care	821
Financials	822
Information Technology	695
Telecommunication Services	46
Utilities	194
Real Estate	252

- Total 4822 10-K filings.

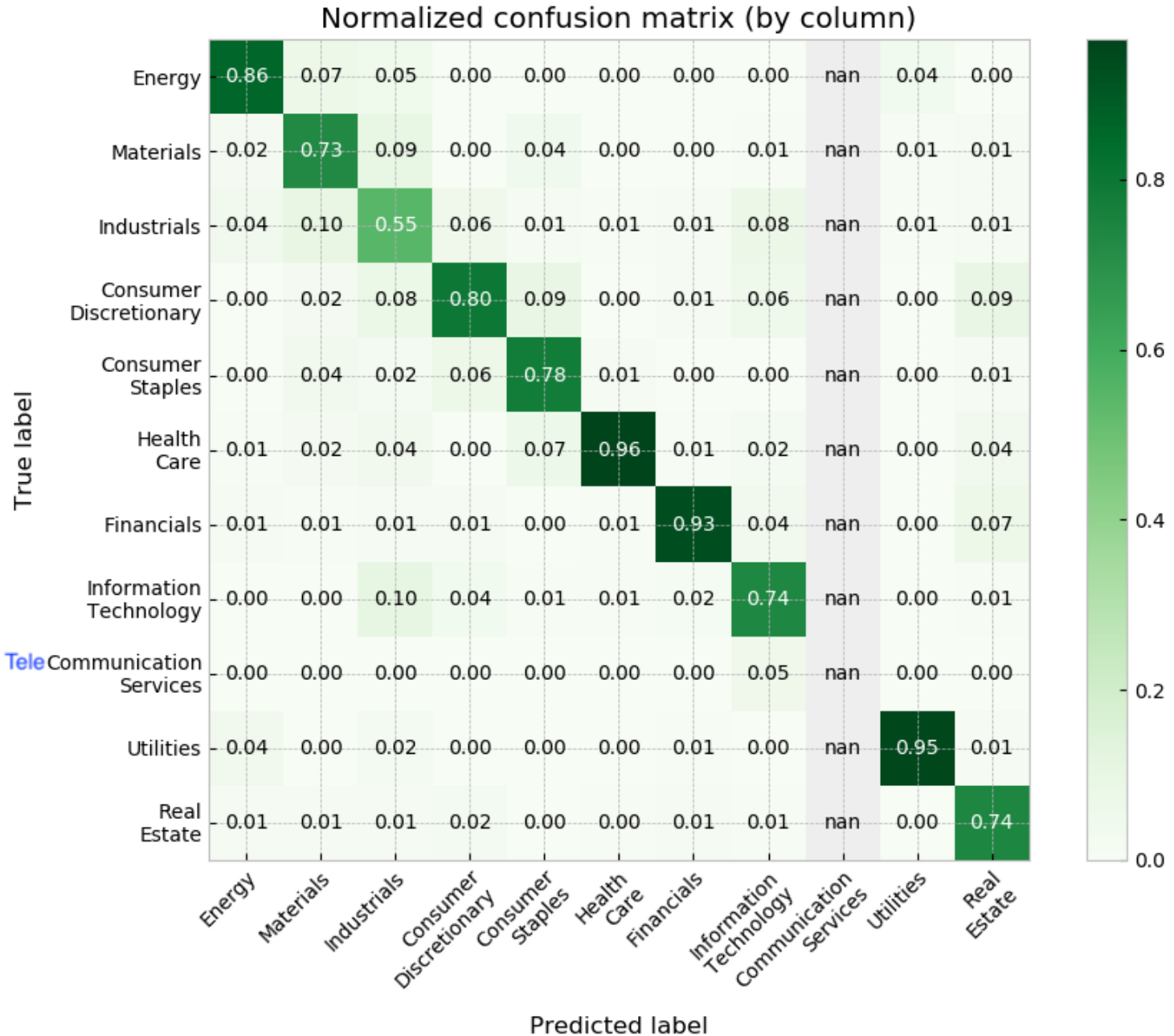
cnn-glove300: Confusion matrix (2016 "test")



cnn-glove300: Confusion matrix normalized by row (2016 "test")



cnn-glove300: Confusion matrix normalized by column (2016 "test")

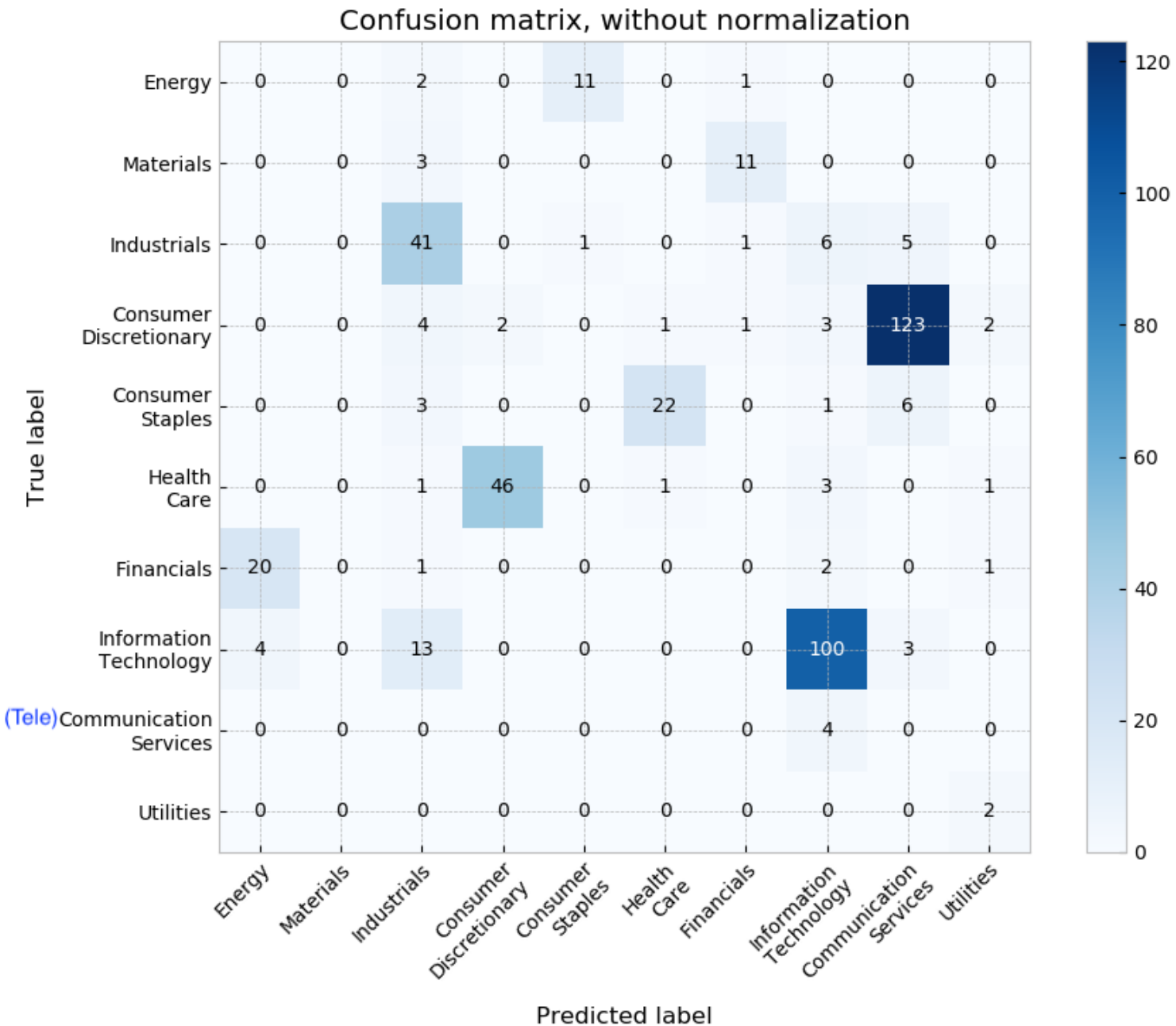


2018 10-K dataset ("test")

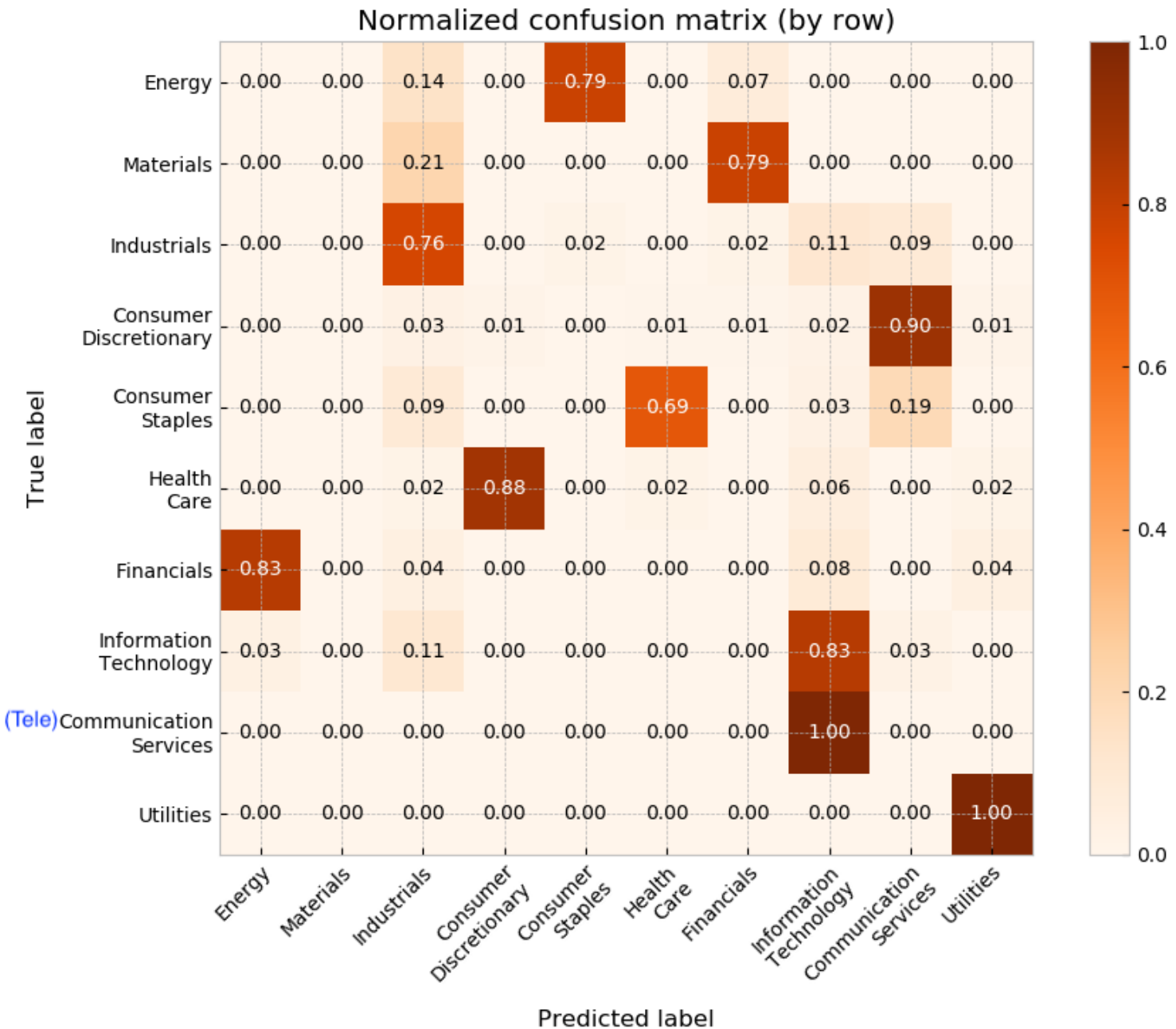
Sectors	Counts
Energy	14
Materials	14
Industrials	54
Consume Discretionary	136
Consumer Staples	32
Health Care	52
Financials	24
Information Technology	120
(Tele)Communication Services	4
Utilities	2
Real Estate	0

- Total 452 10-K filings. (My current list of 10-K file names for 2018 is far from complete, so only a subset of the full 2018 10-K dataset is extracted.)

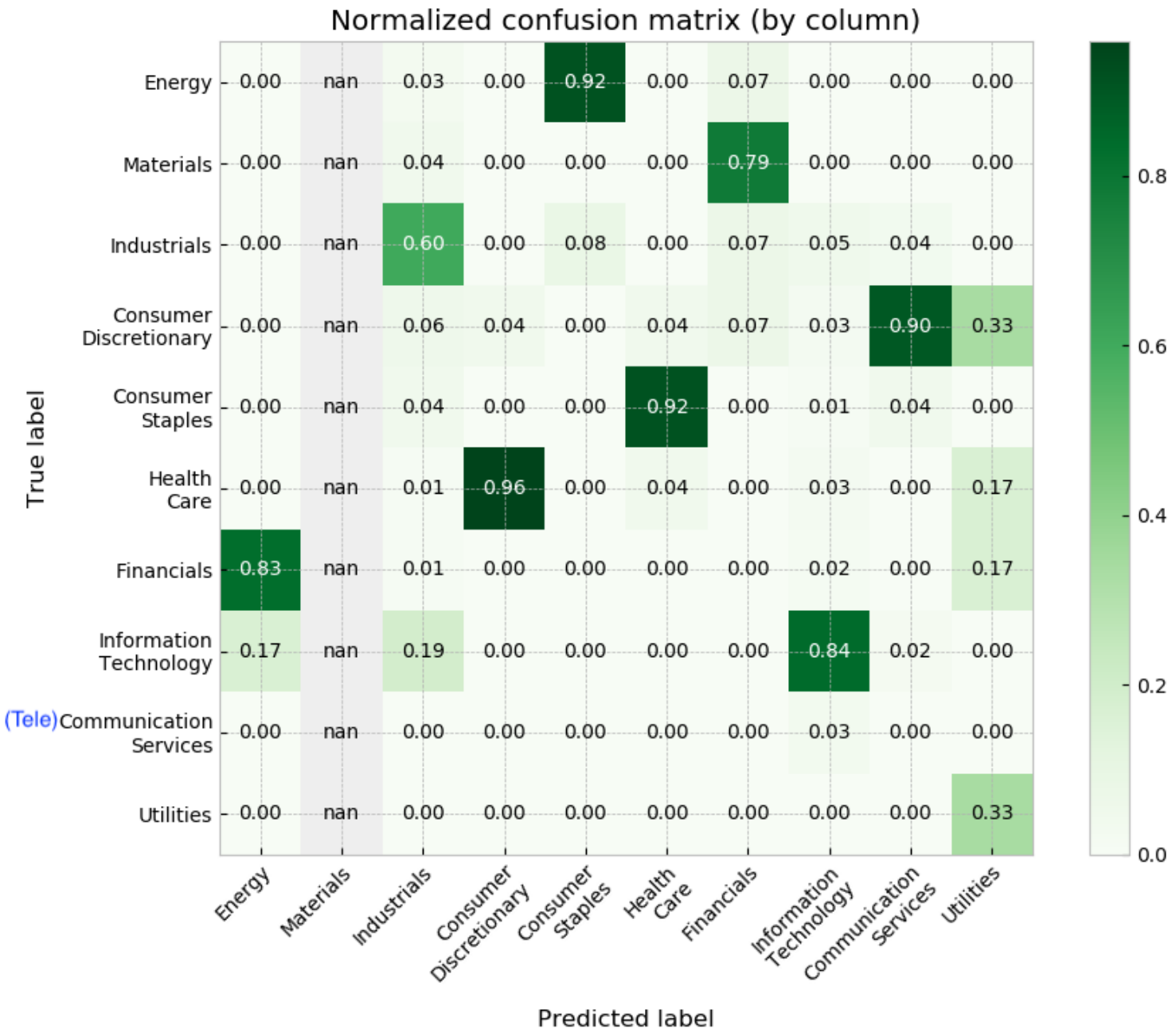
cnn-glove300: Confusion matrix (2018 "test")



cnn-glove300: Confusion matrix normalized by row (2018"test")



cnn-glove300: Confusion matrix normalized by column (2018 "test")



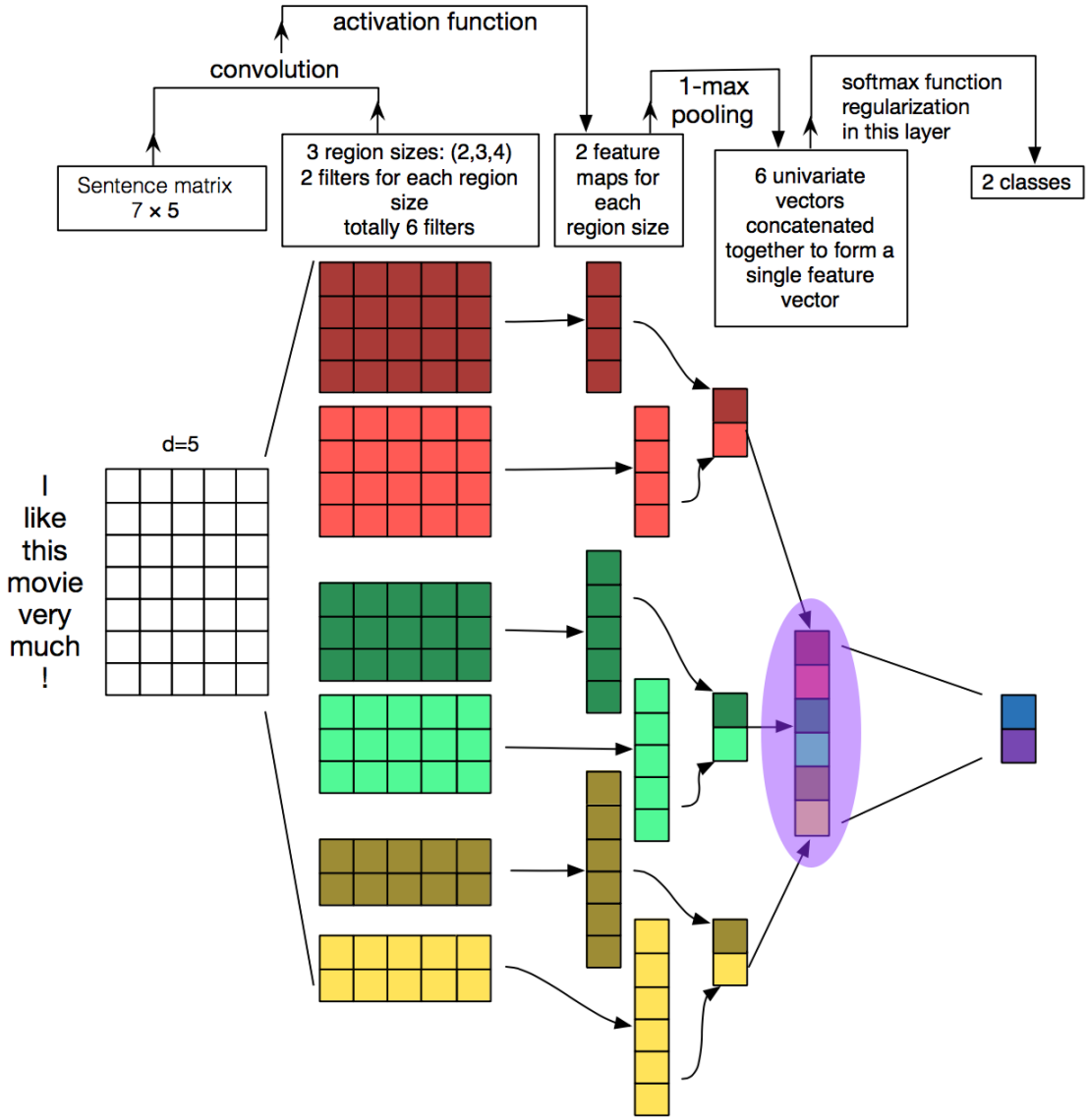
Changes of the GICS on September 28, 2018

Summary of changes

- Telecommunication Services Sector renamed to Communication Services Sector
  - The Telecommunication Services Sector is renamed to Communication Services to include companies that facilitate communication and offer related content through various media. It includes:
  - Media companies moved from Consumer Discretionary to Communication Services
  - Internet services companies moved from Information Technology to Communication Services
- Information Technology Sector
  - The Internet Software & Services Industry and Sub-Industry is discontinued
  - A new Sub-Industry is created under the IT Services Industry called Internet Services & Infrastructure
  - The Application Software Sub-Industry is updated to include cloud-based software companies
- Consumer Discretionary Sector
  - The Media Industry Group is moved out of Consumer Discretionary and into the Communication Services Sector, and renamed Media & Entertainment
  - E-commerce companies are moved from Information Technology to Consumer Discretionary

Where to extract embeddings of 10-K?

Where to extract embeddings of 10-K?



Where to extract embeddings of 10-K?

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 45000)	0
embedding_1 (Embedding)	(None, 45000, 300)	46318800
conv1d_1 (Conv1D)	(None, 44996, 128)	192128
max_pooling1d_1 (MaxPooling1	(None, 8999, 128)	0
conv1d_2 (Conv1D)	(None, 8995, 128)	82048
max_pooling1d_2 (MaxPooling1	(None, 1799, 128)	0
conv1d_3 (Conv1D)	(None, 1795, 128)	82048
global_max_pooling1d_1 (Glob	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dense_2 (Dense)	(None, 11)	1419

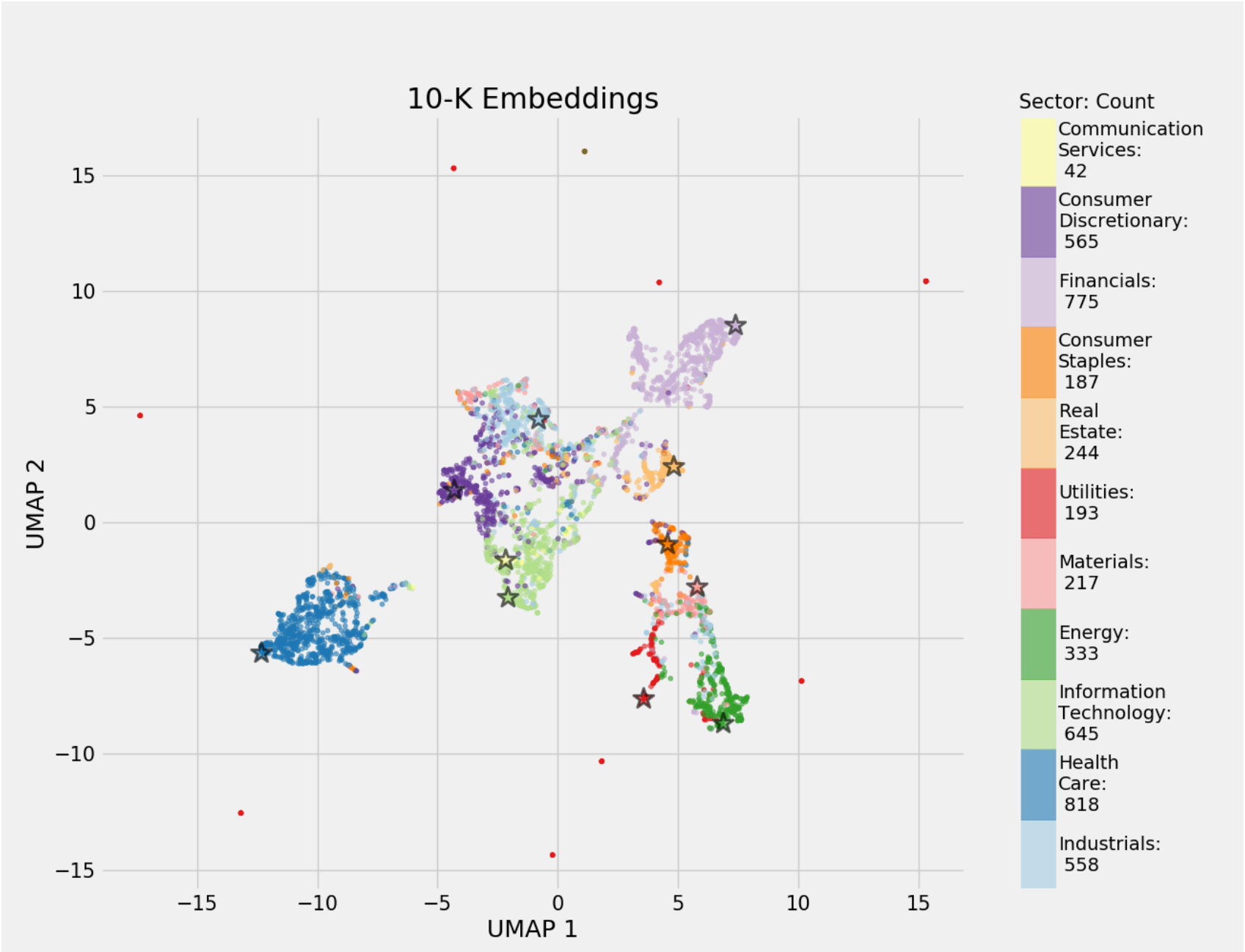
Trainable params: 374,155, Non-trainable params: 46,318,800

Feature vectors, softmax, predictions

- Let $\mathbf{u}$ be the input vector to the prediction layer of the text CNN classifier. We will extract $\mathbf{u}$ as the embedding or feature vector for a 10-K text.
- Prediction (probability vector): $p = \text{softmax}(\mathbf{W}\mathbf{u} + \mathbf{b})$, where $\mathbf{W}$ and $\mathbf{b}$ are parameters of the prediction layer.
- Softmax function:

$$\sigma(\mathbf{z})_k = \frac{e^{z_k}}{\sum_{k=1}^{K} e^{z_k}}, \quad \mathbf{z} \in \mathbb{R}^K$$

cnn-glove300 10-K embeddings (2017)



What are the stars?

What are the stars?

They are the 11 row vectors of the parameter $\mathbf{W}$ in the prediction layer of the text CNN classifier.

Clustering 10-K using the Louvian method

1. Represent a 10-K text as a vector $\mathbf{x_i}$.
2. Compute similarity matrix: $S_{ij} = \exp(-\gamma \|\mathbf{x_i} - \mathbf{x_j}\|^2), \gamma > 0$.
   (Here we used the radial basis function (rbf) kernel.)
3. Threshold the similarity matrix to make an adjacency matrix:
   $A_{ij} = 1_{S_{ij} \in \text{upperquantile}(q)}, q \in [0, 1]$ is a parameter.
4. Clustering algorithm:

Louvain method [3] is a bottom-up, agglomerative algorithm for community detection. It is a greedy, iterative two-phase algorithm. First small communities are found by *maximizing modularity locally* on all nodes, then each small community is grouped into one node and the first step is repeated.

Modularity [4] is a benefit function designed to measure the quality of a division of a network into modules. It reflects the concentration of links within modules compared with a randomly wired network having the same number of nodes and node degrees. $Q = \frac{1}{2m} \sum_{k=1}^{K} \sum_{i,j \in c_k} (A_{ij} - \frac{k_i k_j}{2m})$

5. Shuffle the corpus to create a reference system.

Fix the clustering architecture, vary 10-K representations

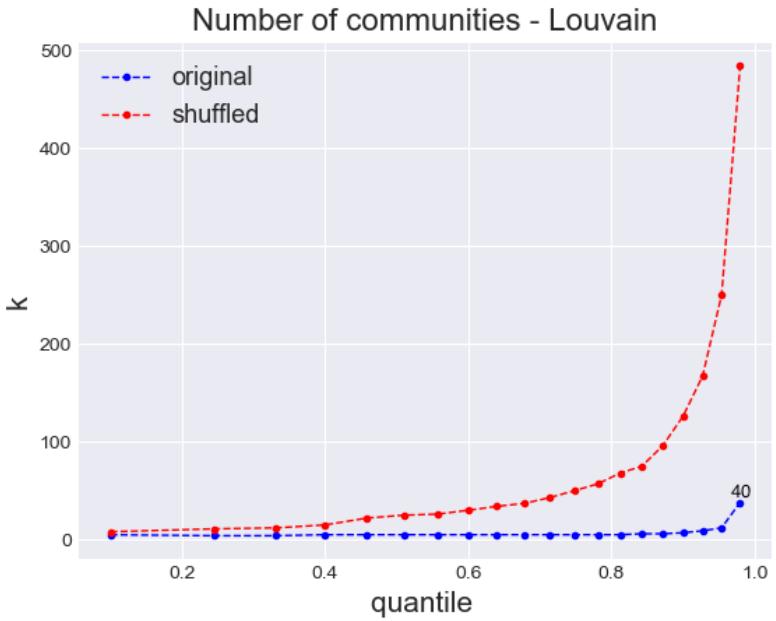
Model	cnn-glove300	bow-tfidf300	avg-glove300
Dimension	128	300	300
Vocabulary	400,000	300	400,000
Weightings	pred_layer input	$tf(w)\log(\frac{1}{df(w)})$	avg(glove300(w))

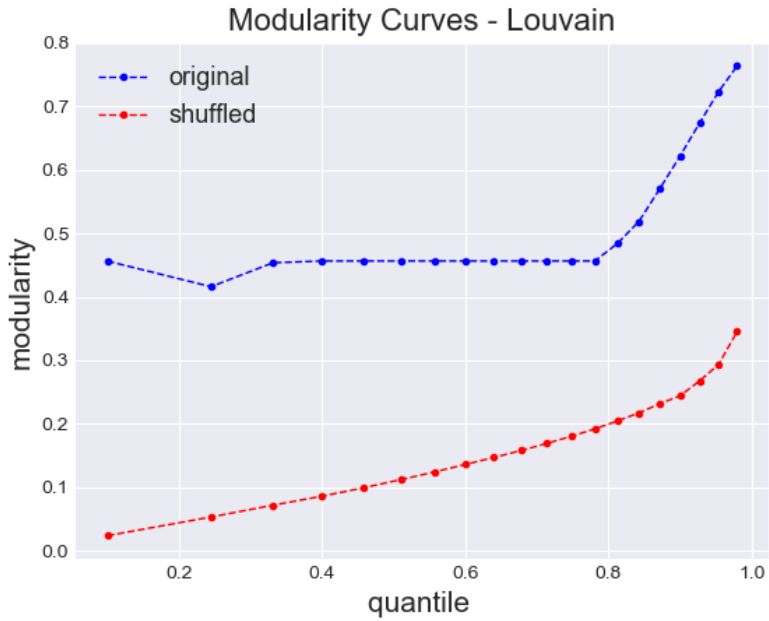
Table: Three different vector representations for 10-K

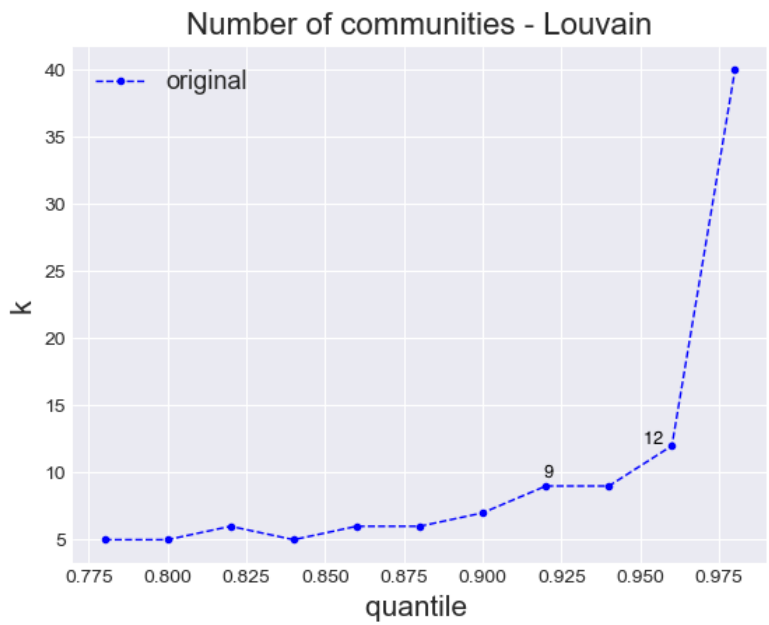
How are the 300 words selected in the model *bow-tfidf300*?

1. Filter the corpus by word-document-frequency: $df(w) \in [0.05, 0.1]$.
2. Take the 300 words with the highest term-frequency in the filtered corpus.

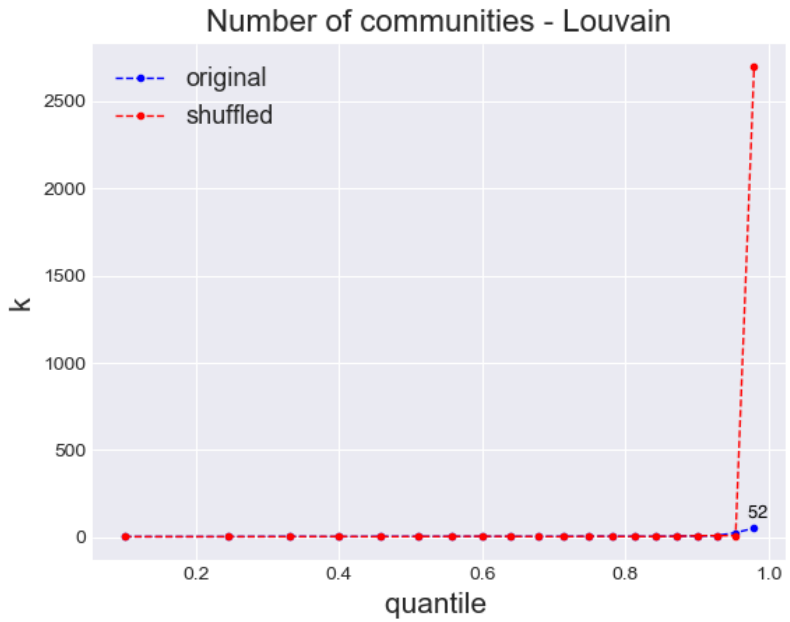
cnn-glove300

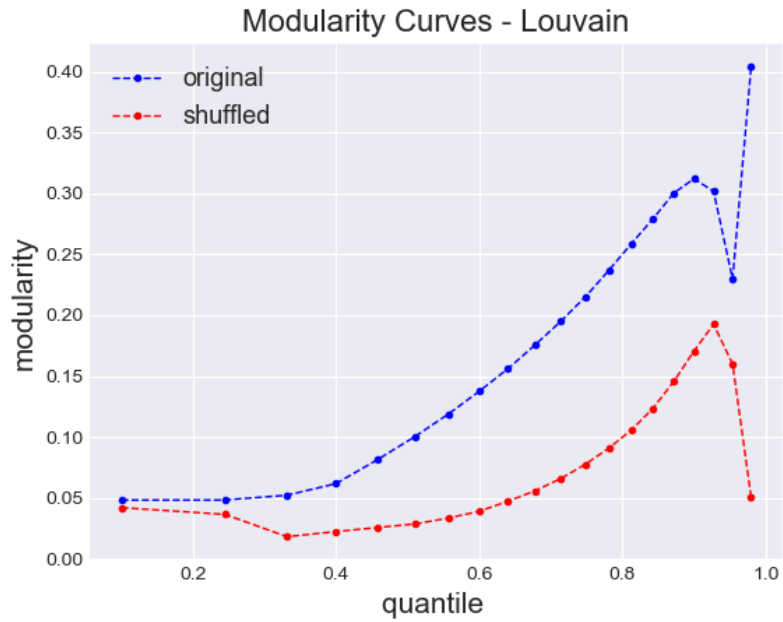


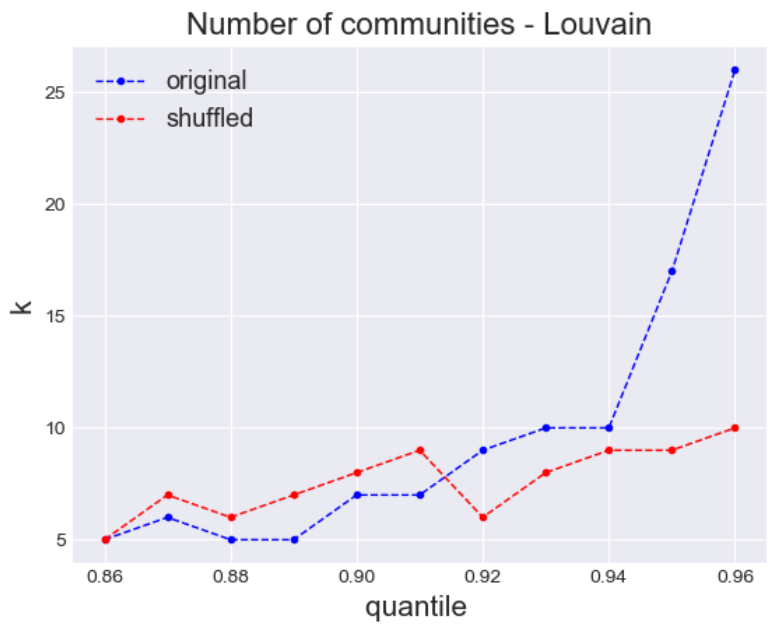




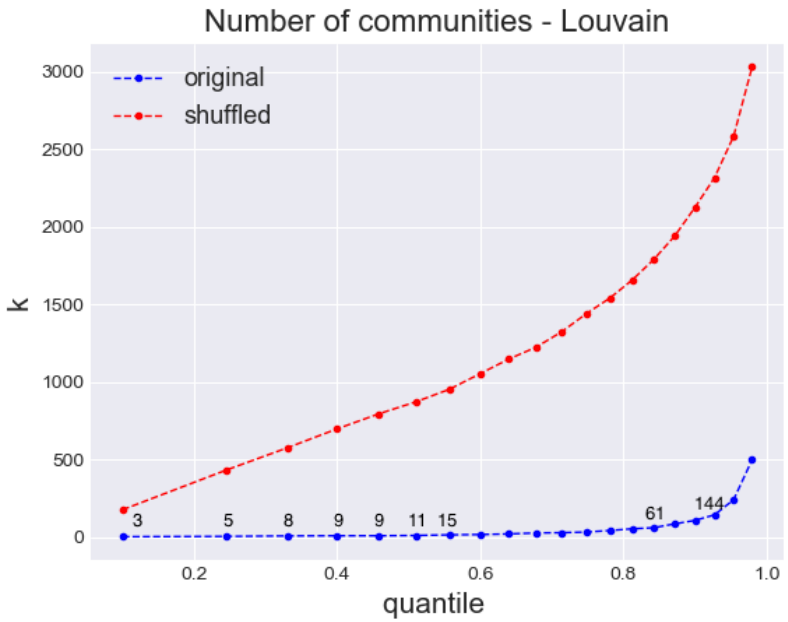
bow-tfidf300 (bag-of-words)

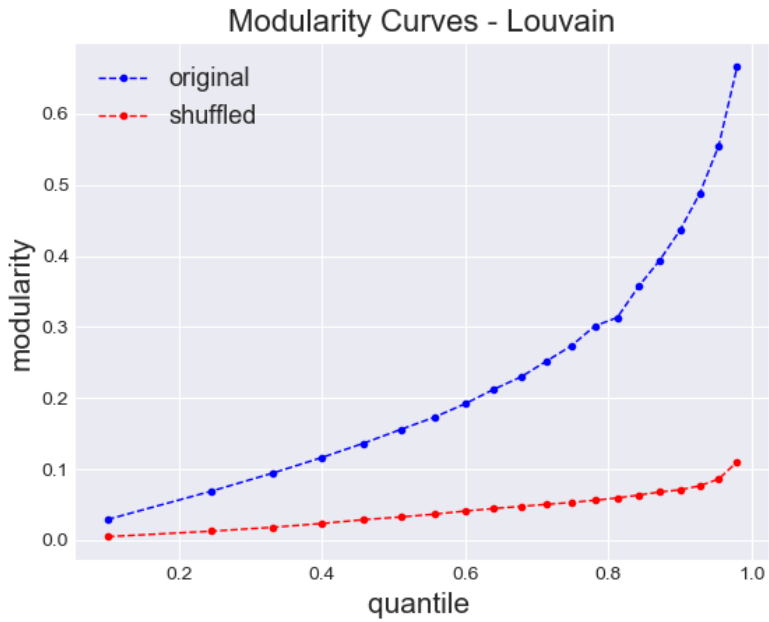


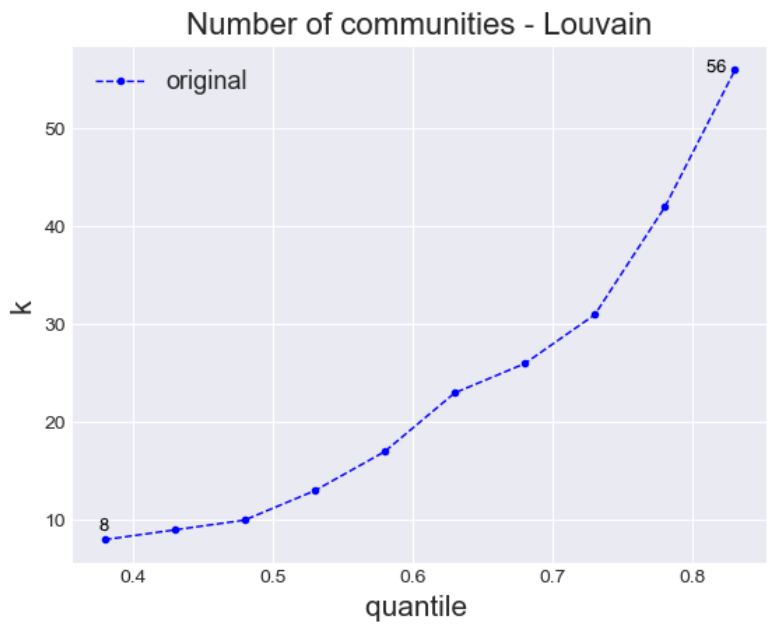




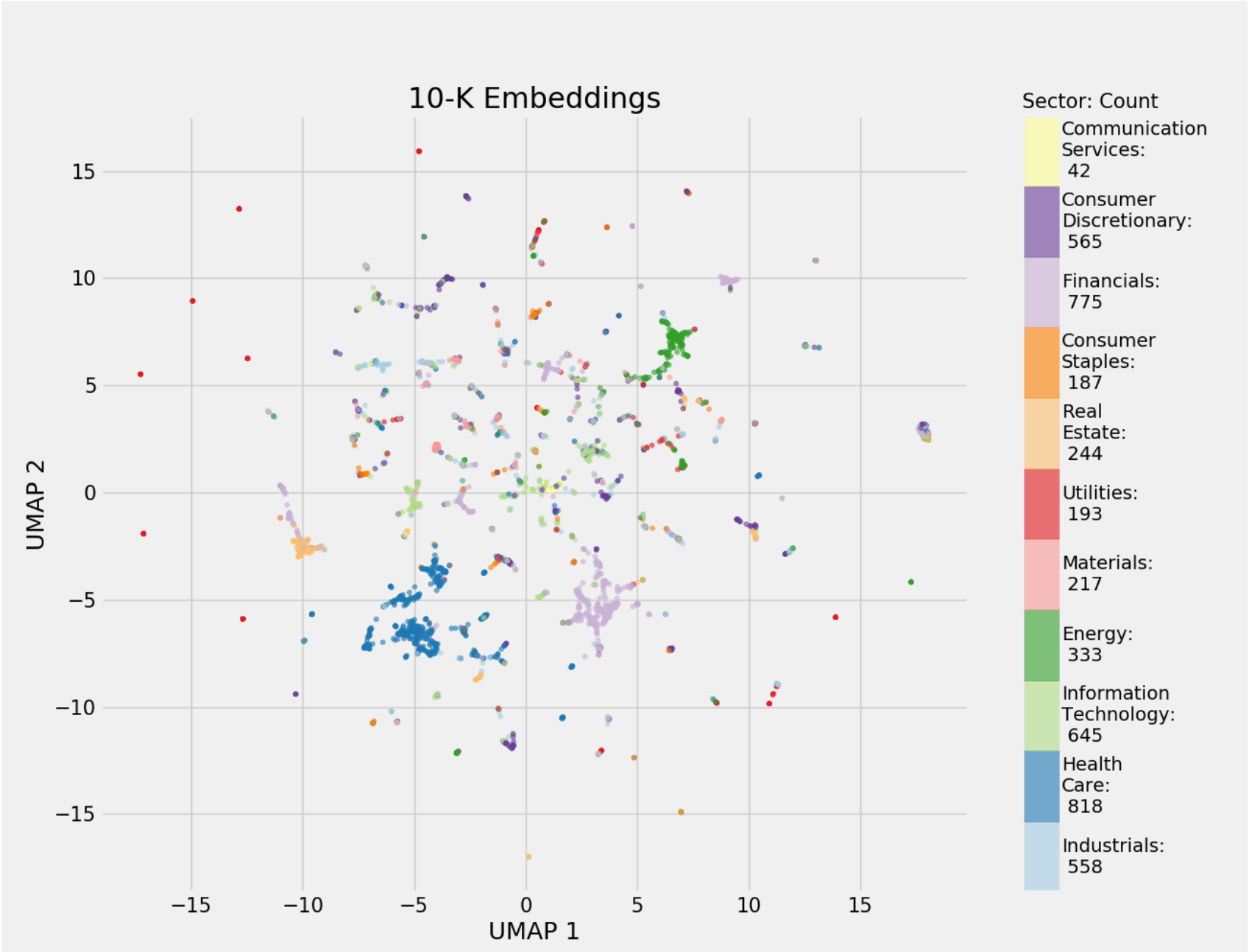
avg-glove300



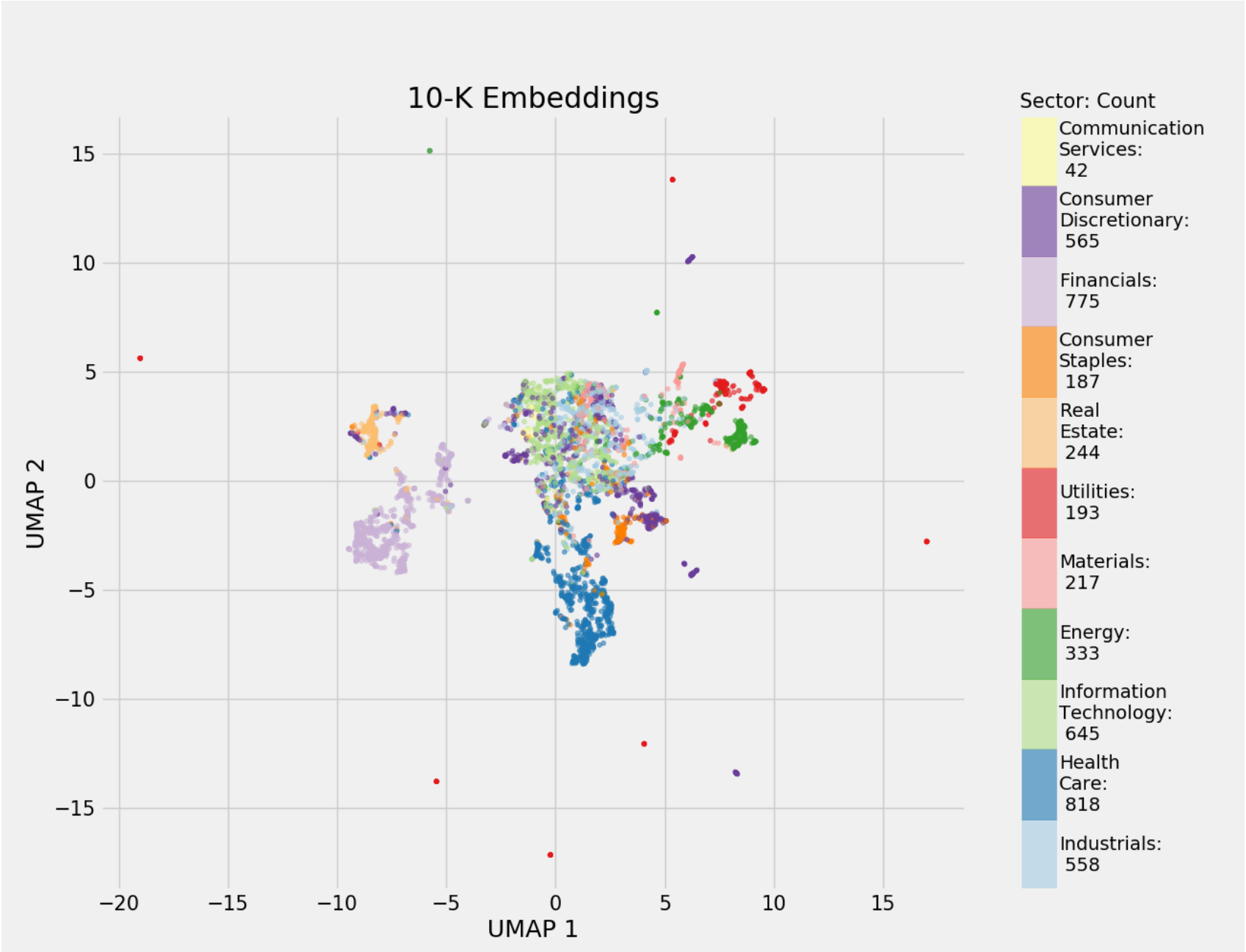




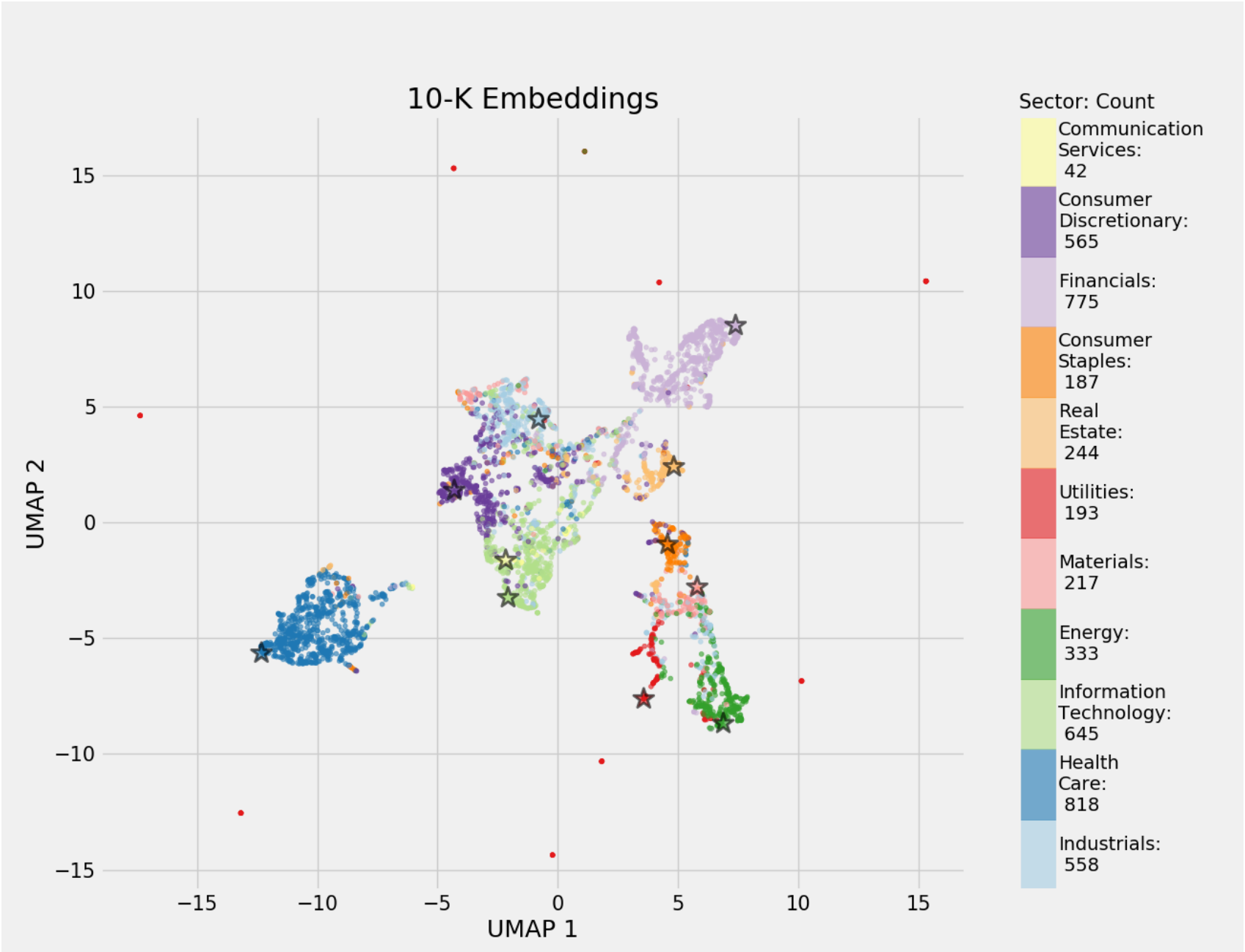
bow-tfidf300 10-K embeddings (2017)



avg-glove300 10-K embeddings (2017)



cnn-glove300 10-K embeddings (2017)



Conclusions and further tasks

- Conclusions:
  1. We demonstrate that pre-trained word vectors are superior feature extractors for the purpose of classifying firms on the basis of their 10-K filings.
  2. The text CNN classifier trained with glove300-word-embeddings (the best) failed, miserably, to recognize the 10-Ks of Tele-communication Services, in particular, confused them mostly with those from Information Technology.
  3. The preliminary results of the 10-K clustering experiments suggests a strong presence of community structure with the number of groups on the order of ten.
- Further tasks:
  1. Compile a more comprehensive 10-K dataset for 2018 and run the test again.
  2. Get a better understanding of the current set of experiments.

References


Kim, Y., 2014. Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014), pp. 1746-1751.


Pennington, J., Socher, R. and Manning, C., 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) pp. 1532-1543.


Blondel, V.D., Guillaume, J.L., Lambiotte, R. and Lefebvre, E., 2008. Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment, 2008(10), p.P10008.


Newman, M.E., 2006. Modularity and community structure in networks. Proceedings of the national academy of sciences, 103(23), pp.8577-8582.