Manuscxript drafts

Gerbrich Ferdinands

1/14/2020

Analysis strategy

Goal: evaluate performance of different models of the ASReview tool.

The software

ASReview takes the following parameters/arguments:

	Configurations
Models	2-Layer Neural Network, Naive Bayes, Random Forest, Support Vector Machine, Logistic Regression
Query Strategies	Cluster Sampling, Maximum Sampling, Cluster * Maximum Sampling, Maximum * Uncertainty Sampling, Maximum * Random Sampling, Cluster * Uncertainty Sampling, Cluster * Random Sampling
Feature extraction strategies	Doc2Vec, TF-IDF, sbert, embeddingIdf

Use these inputs to predict relevance of papers.

Stage 1: hyperparameter optimization

We are going to test 5 models on 5 different datasets.

Datasets

ptsd

ace

hall

nagtegaal - van PhD van Lars

medische van Jan

Models

- Naive Bayes
- Random Forests
- Support Vecor Machine
- Logistic Regression
- Dense Neural Network

Or, more specific:

Models	Feature extraction strategies
dense_nn	doc2vec
nb	tfidf
rf	tfidf
svm	doc2vec
lr	tfidf

The other parameters remain fixed over the 5 models:

- Query Strategy = max
- Balance Strategy = triple
- n_instances=10 (number of papers each query)
- n_prior_included = 5
- n_prior_excluded = 5

Hyperparameters

Every model has its own set of hyperparameters:

Optimization

The hyperparameters are optimized on the 5 datasets in three different ways:

- 1 on 1: maximum performance
- 4 on 1: cross-validation
- 5 on 1: more data = more better?

This results $(5 + 5 + 1) * 5$ sets of hyperparameters.

Stage 2: simulation

for every for every model (5), for every dataset (5) and for every set of optimized hyperparameters (3), a simulation study is performed. From these $5 * 5 * 3 = 75$ simulation studies, performance of the different models is evaluated.

Outcomes

Several metrics are used to compare performance of different models over datasets,

Dataset	Naive Bayes	Random Forests	Support Vector Machine	Logistic Regression	Dense Neural Network
ptsd	?				
ace	?				
hall	?				
nagtegaal	?				
. . . .	?				

? How to compare outcomes of 3 different optimization strategies?

Appendix A - list of definitions

Machine learning algorithms cannot predict the relevance of abstracts from the raw texts as they are. The content of the texts needs to be transformed into numerical representations. The processs of transforming texts to numerical feature vectors is called word embeddings.

A classical example of word embeddings is 'bag of words'. For each each text, the number of occurrences of each word is stored. This leads to n features, where n is the number of distinct words in the texts. (Pedregosa et al. 2011)

Word embeddings allows ASReview to predict relevance of abstracts from the features of abstracts of which relevance is known.

corpus = all the text:

ASReview implements several feature extraction strategies. The following will be compared:

The model is typically a learning algorithm used to predict the relevance of text.

Active learning = increasing classification performance with every query. The query strategy determines the way unlabeled papers are queried to the researcher.

(Danka and Horvath, n.d.)

Feature Extraction Strategies

split_ta = overall hyperparameter

TF-IDF The bag-of-words method is simplistic and will highly value often occuring but otherwise meaningless words such as "and".

Term-frequency Inverse Document Frequency (Ramos and others 2003) circumvents this problem by adjusting a term frequency in a text with the inverse docuement frequency, the frequency of a given word in the entire corpus.

hyperparameters

```
ngram_max: int
        Can use up to ngrams up to ngram_max. For example in the case of
        ngram_max=2, monograms and bigrams could be used.
```

Doc2Vec Predicts words from context. Aims at capturing the relations between word (man-woman, king-queen). (Le and Mikolov 2014). Using a neural network.

using Continuous Bag-of-Words (CBOW), Skip-Gram model, …. Word vector W and extra: document vector D, trained to predict words in the text.

From gensim (Řehůřek and Sojka 2010).

```
Arguments
---------
vector_size: int
    Output size of the vector.
epochs: int
    Number of epochs to train the doc2vec model.
min_count: int
    Minimum number of occurences for a word in the corpus for it to
    be included in the model.
workers: int
    Number of threads to train the model with.
window: int
    Maximum distance over which word vectors influence each other.
dm_concat: int
    Whether to concatenate word vectors or not.
    See paper for more detail.
dm: int
    Model to use.
    0: Use distribute bag of words (DBOW).
    1: Use distributed memory (DM).
    2: Use both of the above with half the vector size and concatenate
    them.
dbow_words: int
    Whether to train the word vectors using the skipgram metho
```

SBERT BERT-base model with mean-tokens pooling (Reimers and Gurevych 2019)

embeddingIdf This model averages the weighted word vectors of all the words in the text, in order to get a single feature vector for each text. The weights are provided by the inverse document frequencies

Models

Naive Bayes Naive Bayes assumes all features are independent given the class value. (Zhang 2004)

ASReview uses the `MultinomialNB` from the scikit-learn package (Pedregosa et al. 2011), that implements the naive Bayes algorithm for multinomially distributed data. `nb`

Hyperparameters

- alpha - accounts for features not present in learning samples and prevents zero probabilities in further computations.

Random Forests A number of decision trees are fit on bootstrapped samples of the original data, (Breiman 2001) RandomForestClassifier from sklearn

Arguments ——— n_estimators: int Number of estimators. max_features: int Number of features in the model. class_weight: float Class weight of the inclusions. random_state: int, RandomState Set the random state of the RNG. """

Support Vector Machine

Logistic Regression

Dense Neural Network

Query Strategies

- Max - Choose the most likely samples to be included according to the model
- Uncertainty - choose the most uncertain samples according to the model (i.e. closest to 0.5 probability) (Lewis and Catlett 1994)
- Random - randomly selects abstracts with no regard to model assigned probabilities.
- Cluster - Use clustering after feature extraction on the dataset. Then the highest probabilities within random clusters are sampled

The following combinations are simulated:

- cluster
- max
- cluster * random
- cluster * uncertainty
- max * cluster
- max * random
- max * uncertainty

Balance Strategies

amount of training data

- n_instances = number of papers queried each query
- n_queries = number of queries
- n_prior_included: 5
- n_prior_excluded:

Combinations

This leads to 119 combinations of configurations.

- Naive bayes only goes with tfidf feature extraction.
- For the feature extraction strategies we will focus on doc2vec and tfidf. (but will compute all 4)
- This leads to 3 * 7 * 4 * 3 + 1 * 7 * 1 * 3 = 273 combinations.

See appendix A for a table containing all 273 combinations.

Performance metrics

Tradeoff: identifying all relevant papers and reducing workload.

What is more important: recall or precision?

Recall more highly valued than precision.

What about class imbalance?

RRF Amount of relevant references found after having screened a certain percentage of the total number of abstracts.

Work saved over sampling (WSS) Indicates how much time can be saved, at a given level of recall. WSS is in terms of the percentage of abstracts that don't have to be screened by the researcher. Typically, WSS is measured at a recall of 0.95.

$$\texttt{WSS} = \frac{TN + FN}{N} - (1 - recall)$$

Raoul

Utility?

F-measure

ROC/AUC

Appendix B - combinations

Model	Query Strategy	Feature extraction strategy
dense_nn	cluster	doc2vec
dense_nn	max	doc2vec
dense_nn	max * cluster	doc2vec
dense_nn	max * uncertainty	doc2vec
dense_nn	max * random	doc2vec
dense_nn	cluster * uncertainty	doc2vec
dense_nn	cluster * random	doc2vec
dense_nn	cluster	tfidf
dense_nn	max	tfidf
dense_nn	max * cluster	tfidf
dense_nn	max * uncertainty	tfidf
dense_nn	max * random	tfidf
dense_nn	cluster * uncertainty	tfidf
dense_nn	cluster * random	tfidf
dense_nn	cluster	sbert
dense_nn	max	sbert

(continued)

Model	Query Strategy	Feature extraction strategy
dense_nn	max * cluster	sbert
dense_nn	max * uncertainty	sbert
dense_nn	max * random	sbert
dense_nn	cluster * uncertainty	sbert
dense_nn	cluster * random	sbert
dense_nn	cluster	embeddingIdf
dense_nn	max	embeddingIdf
dense_nn	max * cluster	embeddingIdf
dense_nn	max * uncertainty	embeddingIdf
dense_nn	max * random	embeddingIdf
dense_nn	cluster * uncertainty	embeddingIdf
dense_nn	cluster * random	embeddingIdf
nb	cluster	tfidf
nb	max	tfidf
nb	max * cluster	tfidf
nb	max * uncertainty	tfidf
nb	max * random	tfidf
nb	cluster * uncertainty	tfidf
nb	cluster * random	tfidf
rf	cluster	doc2vec
rf	max	doc2vec
rf	max * cluster	doc2vec
rf	max * uncertainty	doc2vec
rf	max * random	doc2vec
rf	cluster * uncertainty	doc2vec
rf	cluster * random	doc2vec
rf	cluster	tfidf
rf	max	tfidf
rf	max * cluster	tfidf
rf	max * uncertainty	tfidf
rf	max * random	tfidf
rf	cluster * uncertainty	tfidf
rf	cluster * random	tfidf
rf	cluster	sbert
rf	max	sbert
rf	max * cluster	sbert
rf	max * uncertainty	sbert
rf	max * random	sbert
rf	cluster * uncertainty	sbert
rf	cluster * random	sbert
rf	cluster	embeddingIdf
rf	max	embeddingIdf
rf	max * cluster	embeddingIdf
rf	max * uncertainty	embeddingIdf
rf	max * random	embeddingIdf
rf	cluster * uncertainty	embeddingIdf
rf	cluster * random	embeddingIdf

(continued)

Model	Query Strategy	Feature extraction strategy
svm	cluster	doc2vec
svm	max	doc2vec
svm	max * cluster	doc2vec
svm	max * uncertainty	doc2vec
svm	max * random	doc2vec
svm	cluster * uncertainty	doc2vec
svm	cluster * random	doc2vec
svm	cluster	tfidf
svm	max	tfidf
svm	max * cluster	tfidf
svm	max * uncertainty	tfidf
svm	max * random	tfidf
svm	cluster * uncertainty	tfidf
svm	cluster * random	tfidf
svm	cluster	sbert
svm	max	sbert
svm	max * cluster	sbert
svm	max * uncertainty	sbert
svm	max * random	sbert
svm	cluster * uncertainty	sbert
svm	cluster * random	sbert
svm	cluster	embeddingIdf
svm	max	embeddingIdf
svm	max * cluster	embeddingIdf
svm	max * uncertainty	embeddingIdf
svm	max * random	embeddingIdf
svm	cluster * uncertainty	embeddingIdf
svm	cluster * random	embeddingIdf
lr	cluster	doc2vec
lr	max	doc2vec
lr	max * cluster	doc2vec
lr	max * uncertainty	doc2vec
lr	max * random	doc2vec
lr	cluster * uncertainty	doc2vec
lr	cluster * random	doc2vec
lr	cluster	tfidf
lr	max	tfidf
lr	max * cluster	tfidf
lr	max * uncertainty	tfidf
lr	max * random	tfidf
lr	cluster * uncertainty	tfidf
lr	cluster * random	tfidf
lr	cluster	sbert
lr	max	sbert
lr	max * cluster	sbert
lr	max * uncertainty	sbert
lr	max * random	sbert

(continued)

Model	Query Strategy	Feature extraction strategy
lr	cluster * uncertainty	sbert
lr	cluster * random	sbert
lr	cluster	embeddingIdf
lr	max	embeddingIdf
lr	max * cluster	embeddingIdf
lr	max * uncertainty	embeddingIdf
lr	max * random	embeddingIdf
lr	cluster * uncertainty	embeddingIdf
lr	cluster * random	embeddingIdf

References

Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32. https://doi.org/10.1023/A:1010933404324.

Danka, Tivadar, and Peter Horvath. n.d. "modAL: A Modular Active Learning Framework for Python."

Le, Quoc V., and Tomas Mikolov. 2014. "Distributed Representations of Sentences and Documents." *arXiv:1405.4053 [Cs]*, May. http://arxiv.org/abs/1405.4053.

Lewis, David D., and Jason Catlett. 1994. "Heterogeneous Uncertainty Sampling for Supervised Learning." In *Machine Learning Proceedings 1994*, edited by William W. Cohen and Haym Hirsh, 148–56. San Francisco (CA): Morgan Kaufmann. https://doi.org/10.1016/B978-1-55860-335-6.50026-X.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12: 2825–30.

Ramos, Juan, and others. 2003. "Using Tf-Idf to Determine Word Relevance in Document Queries." In *Proceedings of the First Instructional Conference on Machine Learning*, 242:133–42. Piscataway, NJ.

Reimers, Nils, and Iryna Gurevych. 2019. "Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks." *arXiv:1908.10084 [Cs]*, August. http://arxiv.org/abs/1908.10084.

Řehůřek, Radim, and Petr Sojka. 2010. "Software Framework for Topic Modelling with Large Corpora." In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50. Valletta, Malta: ELRA.

Zhang, Harry. 2004. "The Optimality of Naive Bayes." In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2004*. Vol. 2.