Title: Log Analysis: Topic Modeling applications on fine-features data processing system

Candidate: Davide Napolitano
Supervisors: Prof. Daniele Apiletti, Emanuele Gallo (Reply Data)

Nowadays an ever-increasing number of operations is performed with the help of computer systems. They are everywhere around us and each of them produces a huge quantity of log files for each operation.
In this context, the aim of this project is to study and create an architecture able to read and categorize the logs produced by computer systems based on logs content.
The outcomes of this project form the foundation of a subsequent task of anomaly detection, which is highly requested since lets companies detect malicious attacks or failures and whereby the focus is moved towards the most relevant events.
As previously mentioned, I deal with log files. They contain the sequential and chronological records of the operations done by computer systems. Since new systems are extremely complex, the interaction of their software components causes the registration of a big number of operations.
Accordingly, analyse and gather useful information from this huge amount of data is challenging and it requires many efforts, because both field expertise and cutting-edge technologies are needed.
The starting point of this thesis is the LogHub dataset repository, where several computer system log types are collected. Specifically, for this research, the focus is on logs coming from "HDFS" and "Spark" distributed systems.
One of the issues of this study is the object manipulated: the log. Differently from common texts like books, logs are short texts with many technical words and useless elements like path or web keys. For this reason, the manipulation and the cleaning phases are essential to provide useful information to the adopted models.
Therefore, the first part of the project is the process of making the data more suitable and it is made by several steps. They are principally based on NLP and Regular Expression crafting, which allow to obtain meaningful words from unstructured logs.
Some of these techniques are common when dealing with text mining, like the removal of accents and special characters, the expansion of contractions, the tokenization and the PoS tagging.
In addition, detailed studies have been conducted on log structure. Indeed, several recurring patterns, corresponding to meaningless elements, have been identified and, through their removal, it is possible to trace logs back to similar structures, reducing differences.
These operations regard reformatting logs error, since usually they are divided into multiple lines, the applications of the Camel Case Split, due to Object based nature of the systems, the upper-case split for exceptions and the removal of patterns, like parentheses with their content and variable value association. Moreover, a personal framework is proposed to lemmatize words that, due to their rare proposition in common texts, are not properly tagged and conducted to their corresponding root.
The final steps in the data processing phase are the removal of words depending on their frequency among documents and the production of multi-grams. The last one, provided a sentence, makes mono-gram, bi-gram and tri-gram, while considering to not create elements that contain the same word multiple times or elements with overlapping meaning.
Provided this data processing system, for both "HDFS" and "Spark" three different datasets are made: ORIGIN+MESSAGE, that considers both the origin and the message elements of logs, MESSAGE, that takes only the message part, and SPLIT, a version in the middle that considers both the message

and the origin but for the latter only monograms are created, in order to decrease its weight in the algorithms.
Afterward, the topic models are implemented. Concisely, they consist in annotating documents with thematic information and in grouping together documents that share similar contents.
About the chosen algorithms, the project is split into two branches. The first one investigates standard approaches, based on probabilistic or algebraical operations, where the analysed models are LDA, NMF and LSA. The second branch, instead, follows a modern approach based on Neural Network and Feature Embedding. For this part the proposed algorithms are ProdLDA, BERTopic and, by combining many of their approaches, my custom model called GEAC.
About the training phase, the datasets final structure have to be considered. They are made by few short entries, making the job hard for the models. For this reason, for all the models a very extensive finetuning has been carried out, otherwise wrong or meaningless solutions would have been found.
All models have been tested with the CV coherence score. Inside the world of coherence scores, the CV is the most meaningful to evaluate the goodness of topics since it provides results close to the human ones, the ground truth for this task.
One issue with topic models is about the quality, since the cited models do not always guarantee that each topic is independent from the others. For this reason, I perform an analysis about the similarity between topics by exploiting the spatial representation of each word inside a topic through the Word2Vec feature embedding model. Different possible solutions have been proposed to achieve this result. One aims to keep the model that obtains the highest CV score but, at the same time, that does not have too similar topics. Another solution tries to achieve a good score with a greedy approach: starting from a good result, it decreases the number of topics, depending on the number of similar topics, until all the topics of a model are not too similar. The last proposal, instead, merges similar topics, decreasing as consequence the number of topics and their similarity.
These proposals have been tested with classical models, in order to explore and understand many behaviours. Instead, for the algorithms based on neural network, these solutions have been directly included in the training pipeline, in particular the first one mentioned, since the different training phase, combined with datasets used, requires these reasonings.
As consequence, for this project the above considerations have been necessary. However, the final aim is not to obtain completely dissimilar topics, but topics that have a level of similarity which does not exceed a threshold beyond which a human would consider the different topics as one. About the value of the threshold, it has been fixed to 85%, since it allows to discriminate topics that are close but, at the same time, different in few details.
Thanks to this analysis, I can select a model that has a high CV metric score and that is also optimal for what the supervisor might infer regarding similarity.
For what concerns results, the performances in classical models are good for NMF and LDA, while LSA is not able to provide meaningful topics. Some issues related to these models are the high number of similar topics when the best model has a high number of topics, and their predisposition to find generally better models with a small number of topics, making them unusable in these cases. Therefore, the nature of the datasets, combined with their short composition, makes the creation of topics difficult, in particular about logs whose topic is unique and that are made up of few terms.
The application of Neural Network models introduces several benefits and improvements. ProdLDA seems to be in line or slightly better than LDA, however its utilization is not straightforward. Indeed, it often produces mixtures of topics that do not provide a real meaning, but rather make it more challenging to understand the quality of the models as the metric provides good values due to the fit of topics to multi-sentences.
The introduction of BERT marks a positive turning point. BERTopic allows higher scores, however, it has problems related to its formulation. Indeed, the use of HDBSCAN for clustering involves having at least two elements per group. This is a problem for all logs whose corresponding topic is

uniquely associated to them, that in this implementation are common to be mis-labelled as outliers. To overcome this drawback, I propose two solutions: in the first one I take each outlier as a cluster with unitary size, in this way those elements are kept rather than being discarded. However, with this solution, it may happen, when the model makes an error, to consider independent clusters when an existing topic is suitable. Directly from this consideration, I present the second solution. For those logs considered as outliers, before assigning a new label, I check if a match with found topics exists. This control is done by looking at the similarity, by using the same approach already proposed in other algorithms. Considering all models analysed so far, I present my custom implementation. It is based on the combination of classical models, such as LDA and NMF, and modern approaches, represented by BERT and Word2Vec. Then, I follow a workflow similar to BERTopic, for dimensionality reduction I use an AutoEncoder rather than UMAP, and finally I apply a clustering algorithm to group data. Unlike BERTopic, I use two different types of algorithms: K-Means, a standard model that is based on Euclidean distance, and K-Medoids, a similar model that allows to explore the space through Cosine similarity.
In this personal proposal, the results further improve, both in terms of coherence and predictions. Indeed, until now the models had several problems on creating some clusters and, sometimes, on making the right associations, while now it is possible to obtain correct topics for all logs analysed.
Overall, at the end of this project, a generic log line can be labelled with an almost black box approach that provides optimal results both under human and machine perspectives.

Spark						
	LDA	NMF	LSI	ProdLDA	BERTopic(2)	GEAC
Orig+Mess	0.695 ± 0.035	0.807 ± 0.018	0.719 ± 0.046	0.731 ± 0.019	0.887 ± 0.011	0.955 ± 0.014
Split	0.694 ± 0.023	0.749 ± 0.019	0.671 ± 0.047	0.739 ± 0.021	0.884 ± 0.030	0.924 ± 0.013
Message	0.658 ± 0.009	0.701 ± 0.023	0.540 ± 0.062	0.748 ± 0.026	0.966 ± 0.021	0.971 ± 0.011

HDFS						
	LDA	NMF	LSI	ProdLDA	BERTopic(2)	GEAC
Orig+Mess	0.711 ± 0.024	0.781 ± 0.017	0.759 ± 0.041	0.732 ± 0.032	0.841 ± 0.014	0.951 ± 0.008
Split	0.743 ± 0.018	0.817 ± 0.034	0.693 ± 0.019	0.725 ± 0.008	0.851 ± 0.019	0.941 ± 0.018
Message	0.783 ± 0.019	0.769 ± 0.021	0.636 ± 0.016	0.726 ± 0.016	0.860 ± 0.012	0.958 ± 0.011

Follows an example of topics created. The best two performing algorithms are chosen for each branch.

executor.CoarseGrainedExecutorBackend: Registered signal handlers for [TERM HUP INT]

ORIGIN +MESSAGE	coarse, grain, backend, register, signal, handler, executor_coarse, coarse_grain, grain_executor, executor_backend, backend_register, register_signal, signal_handler, executor_coarse_grain, coarse_grain_executor, grain_executor_backend, executor_backend_register, backend_register_signal, register_signal_handler
SPLIT	coarse, grain, backend, register, signal, handler, register_signal, signal_handler, register_signal_handler
MESSAGE	register, signal, handler, register_signal, signal_handler, register_signal_handler

Version	LDA	NMF	BERTopic	GEAC
ORIGIN + MESSAGE	grain_executor_backend, executor_coarse_grain, backend, coarse_grain_executor, grain_executor, coarse_grain, executor_coarse, executor_backend, coarse, grain	register, backend_register, executor_backend_register,grain, grain_executor_backend, coarse_grain, executor_backend, backend, coarse_grain_executor…	... executor_backend_register, backend_register, driver, register, handler, unknown_driver, driver_disconnect, disconnect...	...register, signal, handler, coarse_grain, backend_register, register_signal, signal_handler, backend_register_signal, register_signal_handler...
SPLIT	store, memory, coarse, backend, grain, start, driver, output, broadcast, task	shutdown, backend, coarse, grain, hook, util, driver, shutdown_hook_call, shutdown_hook, call	… register, connect, connect_driver, handler, driver_disconnect, driver, register_signal_handler, disconnect, register_driver, register_signal, signal, unknown_driver_disconnect, signal_handler...	coarse, grain, backend, register, signal, handler, register_signal, signal_handler, register_signal_handler
MESSAGE	register, block, block_manager, manager, register_block_manager, register_block, try_register_block, try, try_register, block_manager_stop	driver, block, register, block manager, shutdown, manager, byte, register block manager, register block, task stage	signal, register_signal_handler, register_signal, signal_handler, handler, register	register, signal, handler, register_signal, signal_handler, register_signal_handler