NLP for streams in Big Data Environments

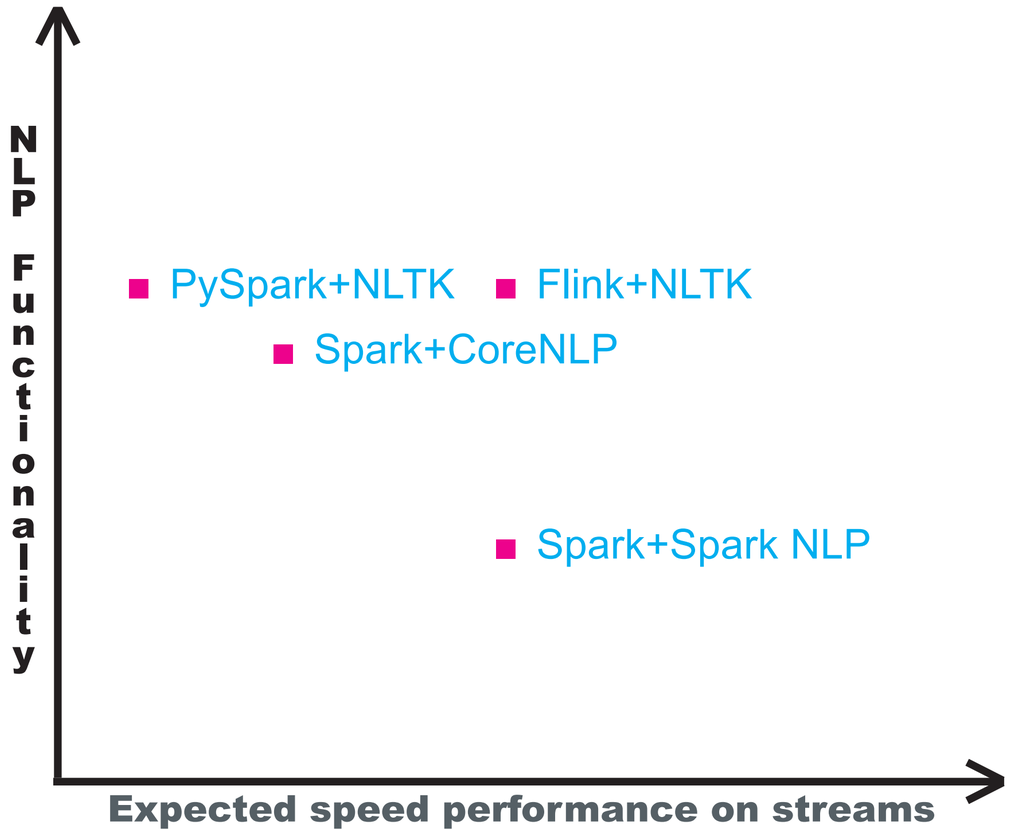
Use cases and technology proposals

Abstract

A recent trend in the wide field of Data Engineering is the transition from batch processing to stream processing. In addition to that, many companies discover more and more use cases for natural language processing (NLP) and the growth speed of unstructured data on the web is increasing. This results in a demand for fast stream processing frameworks that can apply NLP techniques on streams. Here, a couple of possible technology stacks for this problem are sketched and discussed. Apache Flink might become the clear future state-of-the-art for this if its machine learning library gets extended to cover NLP.

Why do we need NLP for big data streams?

- Amazon reviews, comments on social network sites, emails and blog posts - the amount of natural language data is permanently growing - and faster than ever. So is the wish to analyse this data stream.
- Data journalists are interested in e.g. trending topic extraction from Twitter (500 Million tweets per day). The existence of the Newsstream project underlines the need of journalists for a tool to automatically retrieve information from tweets and news sites. However, Newsstream only handles a subset of the daily tweets and is not open source or openly available.
- Xing is working on extracting information from job postings with NLP to improve their job recommendation system. They already have a data stream infrastructure and new job offers could be modelled as additional stream events.
- SAP is researching to further automatize work that is currently done by humans by utilizing NLP to analyse invoices



Overview of possible technology stacks:

- PySpark+NLTK: The Python based NLTK library is arguably the best NLP library in terms of functionality. Spark, however, runs in the JVM, which means that every object must be serialized and copied from the JVM to a Python process in order to be able to be processed by the NLP routines of the whole pipeline.
- Flink+NLTK: Apache Flink supports real-time stream processing, while Spark treats stream as micro-batches. Therefore systems based on Flink have a lower latency for stream processing. However, the FlinkML library so far does not include NLP. It is not clear if a Flink+NLTK system would outperform a Spark+Spark NLP technology stack, due to the communication between the Python process and JVM needed for the Flink + NLTK system.
- Spark+Spark NLP: John Snow Labs NLP is a relatively new library built on top of Apache Spark and its SparkML library. Like Spark itself, it is written in Scala and therefore outperforms combinations of Spark with librarys that run outside the JVM. However, is is not yet as feature-rich as other well established librarys.

Conclusion & Future Work:

- Currently there is no technology stack that can combine the highest possible latency in stream processing, Apache Flink, with a native use of state-of-the-art NLP tools. Future work consists of extending the FlinkML lib to enable Flink to directly deal with natural language processing on streams. This would help data journalists and companies to analyse Twitter or other stream based natural language data.

Sources:

https://spark-packages.org/package/databricks/spark-corenlp
https://dzone.com/articles/apache-flink-vs-apache-spark-brewing-codes
https://flink.apache.org/
http://www.nltk.org/
https://databricks.com/blog/2017/10/19/introducing-natural-language-processing-library-apache-spark.html
https://newsstreamproject.org/
https://blog.microfocus.com/how-much-data-is-created-on-the-internet-each-day/
https://www.talonstorage.com/blog/how-to-address-the-exponential-growth-of-unstructured-data-using-enterprise-data-storage+Lectures

Projektbeteiligte

Adrian Loy - Master Student at Hasso-Plattner Institut

HPI Hasso Plattner Institut
IT Systems Engineering | Universität Potsdam