News Signals: An NLP Library for Text and Time Series

Chris Hokamp* and **Demian Gholipour Ghalandari*** and **Parsa Ghaffari**
Quantexa
<firstname><lastname>@quantexa.com

Abstract

We present an open-source Python library for building and using datasets where inputs are clusters of textual data, and outputs are sequences of real values representing one or more time series signals. The `news-signals` library supports diverse data science and NLP problem settings related to the prediction of time series behaviour using textual data feeds. For example, in the news domain, inputs are document clusters corresponding to daily news articles about a particular entity, and targets are explicitly associated real-valued time series: the volume of news about a particular person or company, or the number of pageviews of specific Wikimedia pages. Despite many industry and research use cases for this class of problem settings, to the best of our knowledge, News Signals is the only open-source library designed specifically to facilitate data science and research settings with natural language inputs and time series targets. In addition to the core codebase for building and interacting with datasets, we also conduct a suite of experiments using several popular Machine Learning libraries, which are used to establish baselines for time series anomaly prediction using textual inputs.

1 Introduction

The natural ordering of many types of data along a time dimension is a consequence of the known physics of our universe. Real-world applications of machine learning often involve data with implicit or explicit temporal ordering. Examples include weather forecasting, market prediction, self-driving cars, and language modeling.

A large body of work on time series forecasting studies models which consume and predict real-valued target signals that are explicitly ordered in time; however, aside from some existing work mainly related to market signal prediction using social media (Chen et al., 2021, 2022; Arno et al., 2022; Li et al., 2014; Bing et al., 2014; Kim et al., 2016; Wang and Luo, 2021), inter alia, the NLP research community has generally not focused on tasks with textual inputs and time series outputs. This is confirmed by the lack of any popular NLP tasks related to time series in result-tracking projects such as nlp-progress[1] and papers-with-code[2].

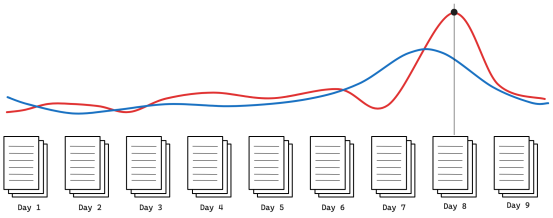

Figure 1: News Signals Datasets: clusters of documents, bucketed by time period, are associated with time series signals. ML models can be trained to predict time series signals using the textual data.

We believe there is potential for novel, impactful research into tasks beyond market signal forecasting, in which textual inputs and real-valued output signals are explicitly organized along a time dimension with fixed-length "ticks". Two reasons for the lack of attention to such tasks to date may be:

1. researchers do not have access to canonical NLP datasets for time series forecasting.

2. data scientists are missing a high level software library for NLP datasets with time series.

Examples of tasks where natural language input can be used to predict a time series signal include:

- weather or pandemic forecasting using social media posts from a recent time period,
- market signal prediction using newsfeeds or bespoke textual data feeds,

*equal contribution

[1] https://nlpprogress.com/
[2] https://paperswithcode.com/

Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023), pages 179–189

- media monitoring for consumer behavior prediction and forecasting,
- forecasting the impact of a news event on the pageviews of a particular website,

and many others. We refer to this general task setting as `text2signal` (T2S).

1.1 `news-signals`

This work introduces `news-signals`[3], a high-level MIT-licensed software package for building and interacting with datasets where inputs are clusters of texts, and outputs are time series signals (Figure 1). Despite the package's news-focused origins, it is built to be a general purpose library for interacting with time-ordered clusters of text and associated time series signals.

Preparing and utilizing datasets for T2S tasks requires purpose-built software for retrieving and sorting data along the time dimension. In many cases, data will be retrieved from one or more APIs, or web-scraped, further complicating dataset generation pipelines. `news-signals` exposes an intuitive interface for generating datasets that we believe will be straightforward for any data scientist or developer familiar with the Python data science software stack (see Section 2).

`news-signals` includes tooling for:

- calling 3rd party APIs to populate signals with text and time series data,
- visualizing signals and associated textual data,
- extending signals with new time series, feeds, and transformations,
- aggregations on textual clusters, such as abstractive and extractive summarization.

`news-signals` provides two primary interfaces: `Signal` and `SignalsDataset`. A `SignalsDataset` represents a collection of related signals. A `Signal` consists of one or more textual **feeds**, each connected to one or more time series. Time series have strictly one real value per-tick, while feeds are time-indexed buckets of textual data. For example, a news signal might contain a feed of all articles from a financial source that mention a particular company, linked to multiple time series representing relevant market signals for that company.

[3] https://github.com/AYLIEN/news-signals-datasets

`news-signals` datasets are designed to be easy to extend with new data sources, entities, and time series signals. In our initial release of the library, we work with three collections of entities: US politicians, NASDAQ-100 companies, and S&P 500 companies (see section 5).

The rest of the paper is organized as follows: section 2 gives an overview of library design and Section 3 describes the `Signal` and `SignalsDataset` APIs, the two main interfaces to time-indexed NLP datasets. Section 4 discusses how datasets can be created. Section 5 describes our example datasets, models, and end-to-end experiments, which are open-source, and can be used as templates for new research projects. Section 6 discusses applications, Section 7 reviews related work, and Section 8 gives conclusions and directions for the future.

2 Time-Indexed NLP Datasets

Traditional NLP and ML datasets consist of iid (X, Y) pairs. These pairs can be assigned indices, and be operated on by standard pre-processing procedures, such as randomly shuffling and splitting into `train`, `dev`, `test` subsets. However, for time series forecasting and related tasks, inputs are ordered along a time axis, and the distribution of later time steps is typically heavily dependent upon the distribution of earlier time steps; therefore, training, dev and test subsets are usually partitioned and split chronologically to reduce the potential for leakage, introducing additional complexity into data preparation.

Within the Python data science ecosystem, libraries such as Numpy (Harris et al., 2020), Pandas (Wes McKinney, 2010), and Pytorch (Paszke et al., 2019) have standardized a syntax for indexing and slicing multi-dimensional matrices and dataframes along axes. When a Pandas dataframe is indexed along a dimension with time-interval semantics, slicing between dates or timestamps is a very useful feature. For example, a user may want to work with the news articles and corresponding time series signals that occurred between particular `START` and `END` dates. Pandas in particular includes rich tooling for indexing and slicing datasets along time-indexed axes, and `news-signals` delegates slice commands and indexing to Pandas, exposing an interface for interacting with datasets

using datetime indices[4].

2.1 `news-signals` Technical Requirements

The key technical desiderata we took into consideration when building `news-signals` are listed below:

- the complexity of data retrieval should be minimized: calling APIs, retrying failed requests, and parsing API output should be invisible to users.
- large datasets containing hundreds or thousands of signals, each lasting for thousands of "ticks", should be straightforward to configure and build.
- standard data science libraries such as Pandas should be used as much as possible to reduce maintenance burden over time.
- transformations on time series such as anomaly detection or trend/seasonality removal should be straightforward to implement.
- the complexity of compressing, saving, and loading datasets locally and remotely should be invisible to users.
- new types of signals should be easy to implement.
- Signals should be easy to use with standard machine learning libraries.

3 The `Signal` and `SignalsDataset` APIs

Signals consist of at least one time series coupled with zero or more textual data feeds. Figure 2 shows an example of creating and populating a `Signal`. Because most functions on the signal class return the signal itself, users can employ a convenient chaining syntax when performing multiple operations on a signal.

The library retrieves and stores the time series and news stories for the signal, and exposes a Pandas-like API to the underlying dataframes. We can add arbitrary textual data feeds to signals; in figure 2, `signal.sample_stories()` samples stories for every day of the time series (see library documentation on GitHub for more detailed information on how this works).

[4] https://pandas.pydata.org/docs/reference/api/pandas.DatetimeIndex.html

```python
import datetime
from news_signals import signals

# wikidata QID for Twitter
qid = 'Q918'

signal = signals.AylienSignal(
    name='Twitter-Signal',
    params={"entity_ids": [qid]}
)

start = '2023-01-01'
end = '2023-06-01'
# retrieve a timeseries for the count of
# news articles per-day for this signal
signal = \
    signal(start, end).anomaly_signal()
# sample stories for every day in the signal
signal = signal.sample_stories()

# let's have a look at the biggest anomaly
top_day = signal.anomalies.idxmax()

# what was going on that day?
stories = signal.feeds_df.loc[top_day]['stories']
for s in stories:
    print(s['title'])

# Twitter experiencing outages nationwide
# Twitter experiencing international outages ...
# It's Not Just You, Twitter Is Acting Weird
# : Twitter briefly goes down
# Twitter outage: what happened, ...
#....
```

Figure 2: Creating and using a news signal

Once feeds and time series have been initialized, users can perform exploratory data analysis (EDA) in many ways, for example by examining and summarizing the news stories for an anomalous window of the signal's time series, or by plotting the signal.

Signals can also be easily mapped into a single dataframe representation by using the `.df` property. Signals' dataframe representations contain the textual and time series data associated with a signal, indexed along a `DatetimeIndex`, but they do not contain metadata such as how the signal is populated from one or more APIs, and transformation semantics such as how anomalies are computed.

Signals automatically differentiate between textual data and time series data types – for example, when `signal.plot()` is called, a signal's associated time series are automatically plotted in a multi-line plot.

3.1 API integrations

Most signals require retrieving data from one or more third-party APIs or on-disk datasets. In the current version of `news-signals`, we provide a deep integration with the Aylien NewsAPI, and additionally implement an interface to the Wikidata pageviews API for building pageview time series for Wikidata items [5].

3.2 The `SignalsDataset` API

Individual signals can be grouped into *datasets*. The `SignalsDataset` is a useful abstraction for working with groups of related signals — concretely, these might be signals for all politicians from a particular country, or for all companies connected to a specific market subset, such as the NASDAQ-100 or the S&P-500. Another dataset type could contain signals encapsulating content and time series related to different social media forums, such as Subreddits (Wang and Luo, 2021). The number of signals in a dataset can easily number in the hundreds or thousands, so we design a simple configuration DSL using `yaml` to allow easy construction of large datasets, which is documented in our GitHub repository.

Aylien NewsAPI and Wikimedia APIs Because our production use cases for `news-signals` are focused upon analyzing news data from the Aylien NewsAPI[6], the flagship Signal type in `news-signals` is currently[7] the `AylienSignal`. This signal type abstracts away API call semantics, allowing users to populate a signal by simply calling `signal(start_date, end_date)`. Of the data sources currently implemented in `news-signals`, Wikidata is completely free, but the Aylien NewsAPI requires a license key. However, we note that the Aylien NewsAPI currently has a two-week free trial allowing significant free API calls[8], and we hope to implement Signal types for fully public data sources beyond Wikidata in the near future.

3.2.1 Saving and loading Datasets

Local and remote serialization and persistence are essential features for dataset-focused libraries, and both Signal and SignalsDataset support saving and loading. We have also implemented persistent on Google Drive and Google Cloud Storage, that only require a remote path to be provided. Datasets are decompressed and cached locally so that the same dataset will not be re-downloaded if it is already available locally.

Library Documentation Section 3 has given only a small sample of the `news-signals` library capabilities, and we refer interested readers to the library documentation on GitHub, which also includes end-to-end example notebooks and video walkthroughs.

4 Building Signals Datasets

As discussed in section 3.2, `news-signals` provides an API for the creation of large-scale datasets representing collections of related signals.

Bootstrapping Datasets using Wikidata The Aylien NewsAPI links named entities in text to their Wikidata IDs (Vrandečić and Krötzsch, 2014). `news-signals` users can make use of the Wikidata Query Service[9] to easily build new datasets starting from SPARQL queries that return sets of matching entities (Prud'hommeaux et al., 2013). We build the datasets for NASDAQ-100, S&P 500, and US Politicians in this manner, and the SPARQL queries used to bootstrap these entity sets are available in our repository. For the purpose of this paper, and to exemplify use of the library, we build three example datasets: NASDAQ100, S&P 500, and US Politicians. Each of these datasets is bootstrapped from a list of Wikidata entities belonging to the respective set. To retrieve the entity sets, we build a SPARQL query returning the set of Wikidata entities that match the query, and then use this entity set to generate a dataset. This is a powerful way to generate arbitrary datasets for collections of related entities: for example, datasets for all politicians from a particular country or all American football players could be generated in this fashion. Note that in some cases Wikidata does not contain all entities in a particular set, for example, the NASDAQ100 dataset contains fewer than 100 entities. Dataset statistics are summarized in Table 1. Each of the entity sets is retrieved via one or more SPARQL queries[10]. We then use the Aylien

[5] https://wikitech.wikimedia.org/wiki/Analytics/AQS/Pageviews

[6] Aylien was acquired by Quantexa in February 2023

[7] as of August 2023

[8] https://aylien.com/news-api-signup

[9] https://query.wikidata.org/

[10] about SPARQL

NewsAPI[11] to sample up to 20 stories about each entity for each day of the time period Jan 2020-Jan 2023.

Multi-document Summarization (MDS) We provide a multi-document summarization model in `news-signals` for turning clusters of news articles associated with a particular timestamp into an easily readable summary. In particular, we use a hybrid extractive-abstractive approach that first uses a centroid-based sentence extraction method (Gholipour Ghalandari, 2017) to select 5 key sentences from the whole collection of provided news articles. We generate an abstractive summary from these sentences using a fine-tuned BART-large model (Lewis et al., 2020). The model was fine-tuned on such extractive summaries on the WCEP dataset (Gholipour Ghalandari et al., 2020), which contains compact event summaries with a neutral style.

Sampling News Data for Entities Importantly, we do not provide all news articles about each entity, rather, we provide only a sample of the news content about the entity for each day. This means that successful models should predict the time-series signal based upon the content of the article, rather than global numerical features [12].

Connecting Entities with Timeseries Signals In our example datasets, we focus upon entities that exist in the Wikidata knowledge graph. Different time series signal sources can be automatically linked to these entities. The Wikimedia API itself exposes several interesting time series signals, such as the number of pageviews and the number of edits for each page. We hypothesize that these signals are affected by events occurring in the real world – when an impactful event connected with an entity occurs, there is likely to be an observable change in signal behavior.

4.1 Dataset Release

To avoid potential licensing issues with releasing the news data content of the example datasets, at this stage we plan to only release the datasets containing article titles instead of full article texts and metadata. We also release a version of the datasets with daily abstractive summaries of the content, which do not reveal any source-specific content or data. Both versions will be available by email request to the authors[13].

Extending NewsSignals Because our datasets are grounded on the Wikidata knowledge graph, they are easy to extend with new inputs, entities, and signals. Obvious extensions to our work might include textual data from platforms such as Twitter and Reddit, and market signals such as stock price or other technical indicators for entities that are connected with publicly traded companies. Datasets should also be easy to extend with additional entities, and we provide a set of tools for extending NewsSignals in the accompanying code repository[14].

4.1.1 Docker Container and Example K8s Configuration

Because `news-signals` is designed to be used in both research and production settings, we have also provided a Dockerfile and an example Kubernetes (K8s) job configuration that can be deployed to Google Cloud Platform with minimal setup required. Together, these assets can be used to build signals datasets at a regular cadence, for example once a day or once a week.

5 Example Models and Experiments

This section presents a suite of example models and experiments for users to quickly adapt to their own task settings, and to verify the utility of `news-signals` by establishing baselines for a straightforward anomaly prediction task.

5.1 Binary Anomaly Prediction Task

In this work, we focus on a simple binary anomaly prediction task, which we treat as text classification. The goal is to predict whether a time series signal about a particular entity is anomalous during some window in the past, present, or future, based on textual information in news feeds about the entity. The input for an individual prediction is a set of news articles, an *aspect* (e.g. an entity) and the target a binary anomaly indicator. For simplicity, we predict the target value of a particular day from the textual input of the same day.

We transform time series signals into binary anomaly predictions with the following procedure:

[11] https://aylien.com/

[12] We may also consider models such as vector auto-regression that use signals derived from textual content as well as real-valued signals

[13] note also that all code used to produce the full datasets is open source

[14] https://github.com/AYLIEN/news-signals-datasets

Dataset Name	Start-End Date	Number of Signals	Total Articles	Time Series Targets
US Politicians	2020-01-01 to 2022-12-31	100	1285238	news volume, Wikimedia pageviews
NASDAQ-100	2020-01-01 to 2022-12-31	99	1569139	news volume, Wikimedia pageviews
S&P-500	2020-01-01 to 2022-12-31	100	1728179	news volume, Wikimedia pageviews

Table 1: Datasets Overview

5.2 Target Signals

We experiment with two different time series target signals: anomalies time series of NewsAPI volume counts and Wikimedia page views. One target time series consists of day-level binary values for the time range of our datasets. We use a simple anomaly detector to convert the raw time series signals into binary values, based on the Z-score: We treat each value x_t in a time series as an anomaly if the following is true:

$$\frac{x_t - \mu}{\sigma} > t \quad (1)$$

where μ is the mean and σ standard deviation of a time series. We set the anomaly threshold t (measured in standard deviations) to 3 which results in a proportion of 1-3% positive examples in our datasets.

5.3 Dataset Splits

Each of the three dataset is split chronologically into training (80%), validation (10%) and test (10%) sections. A trained model is informed about all entities in the training data and is tested to apply this knowledge to future data about these entities. The split can also be done across entities to test whether models can generalize to new entities. In this work, we focus on the simpler setting where the entities are known. Note that this does not apply to the zero-shot baselines using LLMs discussed below.

5.4 Balanced Sampling for Training

We preserve the validation and test dataset split as they are, i.e. with a small amount of 1-3% of positive labels, and as continuous time periods. Since training with this label imbalance results in poor results, we create modified training datasets from the time period of the training split: we randomly sample 10,000 positive and 10,000 negative examples for each dataset.

5.5 Compressing Textual Input

Since we are dealing with a large amount of text for each individual prediction task, i.e. a set of 20 news articles, we need to compress these articles into a shorter text to fit the input size of typical current deep learning models. In our experiments, we use the concatenation of all headlines of a day as the textual input. We leave a comparison to alternatives, e.g. multi-document summaries or representative articles, to future work.

5.6 Models for Anomaly Classification

We include several text classification baselines that predict the target based on one day of compressed textual content:

Fine-tuned Transformer Classifier: We fine-tune the pre-trained RoBERTa-base model (Liu et al., 2019) with an un-trained randomly initialized binary classification head. We fine-tune the model on 1 epoch of the label-balanced training examples with a batch size of 8, a learning rate of 2e-5 and a weight decay of 0.01, using the Adam optimizer.

Random Forest with Sparse Lexical Features: We train random forest models on binary lexical features, to explore how well the target signals are represented in surface-level text. We use `sklearn`[15] to extract sparse binary token-indicator features, with a vocabulary of 10,000 tokens, excluding stop words. We train the models with 100 trees and a maximum depth of 20. We determined these values on the validation datasets.

Zero-Shot Classification with Llama-2 (13B): We use Meta's `Llama-2-13b-chat` [16] model for zero-shot classification. We provide the 20 headlines of a day along with a prompt that describes the target signals as an input. The prompt used in the presented experiments is shown in Appendix B.

[15] https://scikit-learn.org/

[16] https://huggingface.co/meta-llama/Llama-2-13b-chat

5.7 Evaluation and Results

We evaluate the binary anomaly classification task using Precision, Recall and F1-score. We put the results into perspective by comparing them to two random baselines: random-uniform, i.e. randomly classifying each input as an anomaly with a 50% chance, and random-target, where we classify each input as an anomaly with a probability set to the proportion of positive examples in the test set. Table 2 shows the results for anomaly classification for news volume and Wikimedia pageviews as target signals. The trained models achieve above-random f1-scores on most of the dataset-target combinations, and obtain better results than the zero-shot baseline. We discuss the results in more detail in Appendix A. Figure 3 shows an example of predicted anomalies, compared to the ground-truth anomalies defined by the anomaly detection method. The predicted anomalies in this example consistently correspond to a spike of Wikipedia page views on the day or shortly after the day on which the input news stories were published.

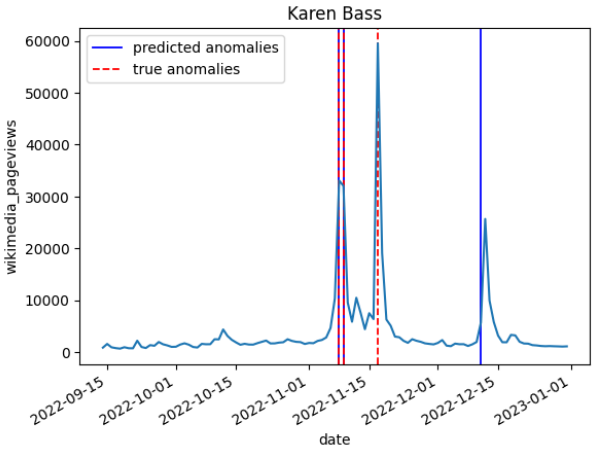

Figure 3: Predicted and ground-truth anomalies of a Wikipedia pageviews time series of US politician Karen Bass. The predictions are from a random forest model with sparse lexical features.

5.8 Extending to forecasting tasks

This experimental setup can easily be converted into forecasting tasks by pairing the text content of a particular day with the target signal shifted by some offset into the future. By sliding our forecasting window earlier than the input, we can also study how well today's news predicts signals that already happened. This may be more relevant for signals that imply significant information asymmetry, such as stock price, as opposed to signals that are public by definition, such as Wikimedia pageviews. Rather than binary anomaly targets, we can train models to directly predict the real-valued signal or quantized representations of the signal.

6 Intended Applications of NewsSignals

Time Series Forecasting using Texual Data As discussed, time series signal forecasting is an important task which is relatively unexplored in the context of models for natural language processing (NLP).

Financial Data Analysis We believe that this dataset and task setting should be straightforward to adapt to financial time series analysis. Financial time series such as stock price and trading volume are impacted by real-world events. The behavior of market signals reflects sentiment about particular entities, and is influenced by events happening in the world. However, market signals may contain opaque and confounding factors that make accurate prediction more challenging. Although this work deliberately does not consider market signals, it is very straightforward to add market time series such as stock price(s) or trading volume to signals.

NLP for Healthcare The `text2signal` task setting is well-suited to the emerging field of BioNLP or NLP for Healthcare – for example, predicting the number of hospital visits in subsequent months based upon a collection of doctor's notes from preceding months, or forecasting total medical expenditure in subsequent months based upon the content of a doctor's notes.

Sentiment To date, sentiment analysis datasets have been created by human annotation. However, the annotation task is difficult to fully specify, and impossible to scale to real-world volumes of data. An insight is that there are many real world signals that can be considered proxies to sentiment, most obviously market signals, especially when the definition of sentiment is constrained to specific (entity, aspect) pairs. Instead of using model-derived sentiment to forecast time series, market signals can be used as ground-truth proxies to sentiment annotations.

Social Sciences Social scientists may be interested in the tooling we have built around the Wikidata SPARQL endpoint, because `news-signals` allows users to easily build a set of signals connected to any set of Wikidata entities.

Target Signal	**News Volume**											
Model/Dataset	**Nasdaq-100**				**Smp-500**				**US-politicians**			
	prec	rec	f1	%pos	prec	rec	f1	%pos	prec	rec	f1	%pos
Random - uniform	0.01	0.43	0.02	0.5	0.01	0.49	0.02	0.49	0.03	0.5	0.05	0.51
Random - target	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.03	0.04	0.04	0.03
Sparse + RF	**0.19**	0.58	**0.28**	0.04	**0.12**	0.30	**0.18**	0.03	0.20	0.28	0.23	0.04
RoBERTa-base	0.12	**0.69**	0.2	0.08	0.1	**0.52**	0.17	0.05	**0.21**	**0.69**	**0.33**	0.08
Llama-2-13b-chat	0.03	0.71	0.06	0.16	0.03	0.47	0.05	0.1	0.05	0.46	0.1	0.22
Target Signal	**Wikimedia Pageviews**											
Model/Dataset	**Nasdaq-100**				**Smp-500**				**US-politicians**			
	prec	rec	f1	%pos	prec	rec	f1	%pos	prec	rec	f1	%pos
Random - uniform	0.02	0.46	0.03	0.5	0.02	0.52	0.04	0.49	0.02	0.51	0.03	0.5
Random - target	0.01	0.01	0.01	0.02	0.02	0.02	0.02	0.02	0.01	0.01	0.01	0.02
Sparse + RF	0.01	0.09	0.02	0.12	0.02	0.11	0.04	0.11	**0.22**	0.16	0.19	0.01
RoBERTa-base	0.01	0.09	0.03	0.11	0.03	0.19	0.05	0.13	0.18	**0.57**	**0.28**	0.05
Llama-2-13b-chat	**0.02**	**0.21**	**0.04**	0.3	**0.05**	**0.24**	**0.08**	0.24	0.04	0.52	0.07	0.22

Table 2: Evaluation results for anomaly classification experiments. %pos indicates the proportion of positive predicted labels.

In one of our example datasets, we produced a signal for every living US politician present in Wikidata, and we believe that many social scientists will be researching similar specific sets of entities and related time series signals.

This section discusses potential applications for `news-signals` and directions for future work.

Causality News-signals may be useful for NLP researchers working on tasks related to causality, because time series signals are well-suited to causality research. In general, we wish to find out what types of information are likely to impact time series signals. Concretely, we may believe that there is a true causal relationship between news and the edit rate on Wikimedia pages.

7 Related Work

NLP and Time Series Dataset Libraries `news-signals` can be seen as sitting between NLP-focused dataset libraries such as Huggingface Datasets (Lhoest et al., 2021) and time series focused libraries such as GluonTS and KATS (Alexandrov et al., 2019; Jiang et al., 2022). We specifically build tooling for working with datasets with textual inputs and time series outputs, and `news-signals` is complementary to and compatible with other popular NLP and time series libraries.

Granger Causality It is natural to consider whether the content of textual inputs "caused" an observed time series signal behavior. Granger causality (Granger, 1969) is a method of measuring the degree to which one signal may cause another. Marcinkevičs and Vogt (2021) propose a framework for discovering Granger Causality with interpretable neural networks.

Summary graphs (Peters et al., 2017) are a useful way of compressing relationships about Granger causality. Wen et al. (2017) introduce a flexible RNN architecture for time series forecasting. Nourbakhsh and Bang (2019) is a position paper discussing the use of PLMs for anomaly detection on financial data.

Time Series prediction with Textual Inputs As discussed in Section 1, one significant line of work focuses on predicting financial time series using signals derived from text, in particular aggregations of sentiment scores from social media posts (Chen et al., 2021, 2022; Arno et al., 2022; Li et al., 2014; Bing et al., 2014; Kim et al., 2016; Wang and Luo, 2021), inter alia.

PLMs and Transfer Learning Recently, significant work has been done to adapt transformer-based models in particular to time series forecasting tasks with flexible semantics (Wen et al., 2023).

Timeline Summarization from News Corpora A related line of work within the NLP community is constructing timelines of important events from large collections of news focused on long-term topics, e.g. disasters or entities (Martschat and Markert, 2018). The methods for identifying important events often make use of time-series-like signals defined over dates: the number of articles published per day or the number of times the date is mentioned in text (Tran et al., 2013; Ghalandari and Ifrim, 2020).

8 Conclusion

We have presented `news-signals`, an open source library for building and working with NLP datasets that predict time series signals based on textual inputs. We hope that this library can be useful to a broad group of researchers and data scientists in both academic and industry settings. Naturally, we would be very happy for additional contributions from the open source community to further improve the library.

References

Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C. Maddix, Syama Sundar Rangapuram, David Salinas, Jasper Schulz, Lorenzo Stella, Ali Caner Türkmen, and Yuyang Wang. 2019. Gluonts: Probabilistic time series models in python. *CoRR*, abs/1906.05264.

Henri Arno, Klaas Mulier, Joke Baeck, and Thomas Demeester. 2022. Next-year bankruptcy prediction from textual data: Benchmark and baselines. In *Proceedings of the Fourth Workshop on Financial Technology and Natural Language Processing (FinNLP)*, pages 187–195, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Li Bing, Keith C. C. Chan, and Carol Ou. 2014. Public sentiment analysis in twitter data for prediction of a company's stock price movements. In *Proceedings of the 2014 IEEE 11th International Conference on E-Business Engineering*, ICEBE '14, page 232–239, USA. IEEE Computer Society.

Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen, editors. 2021. *Proceedings of the Third Workshop on Financial Technology and Natural Language Processing*. -, Online.

Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen, editors. 2022. *Proceedings of the Fourth Workshop on Financial Technology and Natural Language Processing (FinNLP)*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid).

Demian Gholipour Ghalandari and Georgiana Ifrim. 2020. Examining the state-of-the-art in news timeline summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1322–1334.

Demian Gholipour Ghalandari. 2017. Revisiting the centroid-based method: A strong baseline for multi-document summarization. *EMNLP 2017*, page 85.

Demian Gholipour Ghalandari, Chris Hokamp, Nghia The Pham, John Glover, and Georgiana Ifrim. 2020. A large-scale multi-document summarization dataset from the Wikipedia current events portal. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1302–1308, Online. Association for Computational Linguistics.

C W J Granger. 1969. Investigating Causal Relations by Econometric Models and Cross-Spectral Methods. *Econometrica*, 37(3):424–438.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature*, 585(7825):357–362.

Xiaodong Jiang, Sudeep Srivastava, Sourav Chatterjee, Yang Yu, Jeffrey Handler, Peiyi Zhang, Rohan Bopardikar, Dawei Li, Yanjun Lin, Uttam Thakore, Michael Brundage, Ginger Holt, Caner Komurlu, Rakshita Nagalla, Zhichao Wang, Hechao Sun, Peng Gao, Wei Cheung, Jun Gao, Qi Wang, Marius Guerard, Morteza Kazemi, Yulin Chen, Chong Zhou, Sean Lee, Nikolay Laptev, Tihamér Levendovszky, Jake Taylor, Huijun Qian, Jian Zhang, Aida Shoydokova, Trisha Singh, Chengjun Zhu, Zeynep Baz, Christoph Bergmeir, Di Yu, Ahmet Koylan, Kun Jiang, Ploy Temiyasathit, and Emre Yurtbay. 2022. Kats.

Young Bin Kim, Jun Gi Kim, Wook Kim, Jae Ho Im, Tae Hyeong Kim, Shin Jin Kang, and Chang Hun Kim. 2016. Predicting fluctuations in cryptocurrency transactions based on user comments and replies. *PLOS ONE*, 11(8):1–17.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online

and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiaodong Li, Haoran Xie, Li Chen, Jianping Wang, and Xiaotie Deng. 2014. News impact on stock price return via sentiment analysis. *Know.-Based Syst.*, 69(1):14–23.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ričards Marcinkevičs and Julia E Vogt. 2021. Interpretable models for granger causality using self-explaining neural networks. In *International Conference on Learning Representations*.

Sebastian Martschat and Katja Markert. 2018. A temporally sensitive submodularity framework for timeline summarization. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 230–240.

Armineh Nourbakhsh and Grace Bang. 2019. A framework for anomaly detection using language modeling, and its applications to finance.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. 2017. *Elements of Causal Inference: Foundations and Learning Algorithms*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA.

Eric Prud'hommeaux, Steve Harris, and Andy Seaborne. 2013. SPARQL 1.1 Query Language. Technical report, W3C.

Giang Bihn Tran, Mohammad Alrifai, and Dat Quoc Nguyen. 2013. Predicting relevant news events for timeline summaries. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 91–92. ACM.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Charlie Wang and Ben Luo. 2021. Predicting $GME stock price movement using sentiment from Reddit r/wallstreetbets. In *Proceedings of the Third Workshop on Financial Technology and Natural Language Processing*, pages 22–30, Online. -.

Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan, and Liang Sun. 2023. Transformers in time series: A survey. In *International Joint Conference on Artificial Intelligence(IJCAI)*.

Ruofeng Wen, Kari Torkkola, Balakrishnan (Murali) Narayanaswamy, and Dhruv Madeka. 2017. A multi-horizon quantile recurrent forecaster. In *NeurIPS 2017*.

Wes McKinney. 2010. Data Structures for Statistical Computing in Python. In *Proceedings of the 9th Python in Science Conference*, pages 56 – 61.

A Discussion of Anomaly Classification Results

The trained models, i.e. RoBERTa-base and the random forest with sparse features achieve considerable improvements over random results on most of the dataset-target combinations, with mixed rankings. In these cases, the models detect 50-70 % of the anomalies while only predicting 3-8% anomalies in total, which is a promising pattern. All baselines show close-to-random results on Nasdaq-100 and Smp-500 with Wikimedia Pageviews. Zero-shot anomaly prediction with `Llama-2-13b-chat` generally performs worse than the trained models, but still better than the random baselines. Our zero-shot approach suffers from over-prediction of the positive class - a behavior that is difficult to tune when designing prompts. We leave more systematic prompt tuning for this task to future work.

B Prompting for Zero-Shot Approach

We use the following prompt template for `LLama-2-13b-chat` to do anomaly classification from news:

Headlines: {{HEADLINES}} The stories above all involve {{ENTITY}} and were published on the same day. Do these news stories indicate one of the most significant events for {{ENTITY}}? Respond with 'no' or 'yes'.

We instantiate the placeholders with headlines and an entity name (person or company) for a specific data item.

We use the following system prompt: *You are an anomaly detector for news.*

We formatted the prompt according to the `LLama-2`-specific pattern.

A key issue in this zero-shot approach is to control the overall proportion of times an anomaly is detected in a dataset, i.e. to express the significance or importance of news stories to entities.

Signal-specific prompts, e.g. directly describing Wikipedia pageviews or news volume, turn out less effective than this generic description.