CrowdFlow: A Cryptocurrency Sentiment Analysis Application

Yanick Bedel (8424886), *Yannick* Königstein (9502377), *Carlo* Rinderer (1902925), *Niklas* Seither (4253802), and *David* Simon (1893552)

[1] *DHBW-Mannheim course WWI22DSA Natural Language Processing*

Abstract.
Investing money always comes with a risk. But especially assets like cryptocurrencies which don't have any material value supporting them carry such uncertainty. This leads to their performance being heavily influenced by how people currently feel about the topic. In this piece of work natural language classifiers like Support Vector Machines and Random Decision Forrests, clustering algortihms like k-means were used next to deep learning methods like OLLAMA and FinBERT in order to correctly label crypto related news' sentiment. The results give insights into the effects of different preprocessing approaches and data aquisition techniques while scoring the greatest accuracy for trinary classification with 0.55 using a FinBERT model and 0.68 for binary classification using a Support Vector Machine.

1 Introduction

Analyzing the sentiment in blog posts offers investors additional valuable insights into the market and the people at a particular time. Since cryptocurrencies don't have an intrinsic value connected to the reality opposed to assets like company stocks, the peoples sentiment can have drastic influence on their performance.

To achieve visual analytics of the connection of the current sentiment towards specific cryptocurrencies and the people's current sentiment towards them we built an application which retrieves the financial data as well as relevant news articles which are classified based on their sentiment.

2 Related Work

There is related work that also dealt with a Cryptocurrency Sentiment Analysis [1]. The authors used the same dataset as we did. The Paper states a study comparing the effectiveness of advanced NLP models, including GPT-4, BERT, and FinBERT, in cryptocurrency sentiment analysis. Their research illustrated the potential of fine-tuned language models to achieve high accuracy in sentiment classification, with the fine-tuned GPT-4 model outperforming other approaches. Furthermore, the value of leveraging domain-specific fine-tuning to enhance model performance in capturing nuanced sentiment dynamics within the cryptocurrency market is demonstrated.

Furthermore, [2] conducted sentiment analysis of cryptocurrency-related tweets using the Support Vector Machine (SVM) algorithm. They analyzed a dataset of tweets about Bitcoin collected in June 2022. They worked in different steps including preprocessing, data labeling, and text transformation using TF-IDF for feature extraction. To address class imbalances in the data, they applied undersampling, oversampling, and a control method without imbalance handling.

Another related work that does sentiment and emotion analysis of cryptocurrency-related tweets proposes an ensemble model, LSTM-GRU, which combines Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) networks in a stacked architecture. A dataset of 40,000 tweets was preprocessed using techniques like stemming and stop-word removal, and the sentiment and emotion labels were assigned using the TextBlob and Text2Emotion libraries. The study evaluated some feature extraction methods, like Bag of Words (BoW), (TF-IDF), and Word2Vec. It turned out that SVM and Logistic Regression performed best with BoW, achieving 90% accuracy in detecting emotions. The LSTM-GRU model achieved a 99% accuracy in sentiment analysis and 92% in emotion detection [3].

3 Methodology

3.1 Preprocessing

Before training the models, we transform the data into a clean, model-compatible format for sentiment analysis by providing our preprocessing components as object-oriented modules. This design choice allows the models to be seamlessly integrated and reuse the techniques within the respective pipelines. By carefully cleaning and standardizing the data at this early stage, we help ensure the robust training of our sentiment classifiers.

The complete preprocessing pipeline generally consists of the following components:

- *Clensing*
- *Tokenizing*
- *Lemmatizing*
- *Vectorizing*

3.1.1 Clensing, Tokenizing and Lemmatizing

The process begins with a custom transformer called CompoundSplitter, which identifies and splits text at special characters or capital letters, replacing them with whitespace to produce a cleaned string. For example, it separates combined words and hashtags into individual words, such as converting "SAP-System" into "SAP System" or "#BitcoinRocks" into "Bitcoin Rocks," thereby reducing ambiguities as well as the complexity of the vocabulary.

After that code sequences from different encodings are removed which are wrongfully displayed in order to reduce possible interruptions in the sentences which could have an adverse effect on the sentiment analysis.

To further refine the preprocessing, we implemented a custom filter targeting words that represent specific assets (e.g., "Bitcoin," "BTC," "ETH") or Fortune 500 company names. These terms, which could introduce bias by being strongly correlated with particular sentiments over time, were replaced with a unified placeholder ("TICKER"). This ensures the machine focuses on linguistic patterns rather than sentiment linked to specific entities.

Additionally, a custom module was introduced to remove hyperlinks, addressing another source of potential noise in the data.

Following these prior methods the NLTKTokenizer is applied to seperate sentences into seperate terms at the whitespace. This task is then followed by the build in function for filtering out stopwords simplifying the vocabulary by removing unmeaningfull and potentially disturbing text sequences.

Beyond these steps, lemmatization is employed to reduce inflected forms of words to their base lemma. This allows the model to treat morphologically related tokens (e.g., "sell," "sells," "selling") as equivalent, which is particularly valuable in financial contexts. By consolidating variations of words, the process reduces redundancy and enhances the consistency of input data. This method is paired with a POS tagger which will only pass those words through the lemmatizer which it can identify as a certain word group.

3.1.2 Vectorizing

After the text has been cleaned and split into tokens, it is transformed into numeric features as model input. Two primary approaches are employed for this transformation: the TF-IDF-Vectorizer and Word2Vec embeddings (using the pre-trained GoogleNews-vectors-negative300) [4].

The TF-IDF-Vectorizer assigns weights to terms based on their frequency and relevance within the corpus, making it particularly effective at identifying domain-specific terms that appear frequently in certain posts but remain rare across the broader dataset. In contrast, Word2Vec captures distributed word representations, mapping words into a continuous embedding space that reveals deeper semantic relationships and patterns between terms. To further enhance the representation of context while utilizing TF-IDF, Tri-Grams are incorporated by analyzing groups of consecutive words rather than individual tokens. This method adds more contextual nuance, enabling the capture of phrase-level patterns. However, Tri-Grams alone cannot fully represent the semantic connections between words.

Both techniques bring complementary strengths to financial sentiment analysis. While TF-IDF excels at highlighting domain-specific terminology, Word2Vec provides insights into the semantic and relational structure of the text. They address the importance of both distinct phrases and subtle linguistic cues, which are critical for improving classification performance in financial tasks.

3.2 Selected Models

The following models were chosen based on their architectural suitability for text classification tasks and their capacity to handle varying degrees of data complexity. Each approach employs preprocessing steps that aim to improve representational quality and thus classification accuracy.

3.2.1 Clustering

K-Means is an unsupervised learning approach, which clusters data points by iteratively assigning instances to the nearest centroid. This could prove valuable in the scenario of only having access to unlabeled data. Principal Component Analysis (PCA) was introduced to reduce the high-dimensional data resulting from the vectorization to three components, a threshold determined via the elbow method. Before training the model the distribution of the labeled data was visualized which resulted in no clusters being identified as can be seen in Figure 1. Visualising unlabeled also didn't return any results, since the need to label the clusters afterwards also didn't provide any reliable solution. For this reason, the clustering model was not further pursued or evaluated.

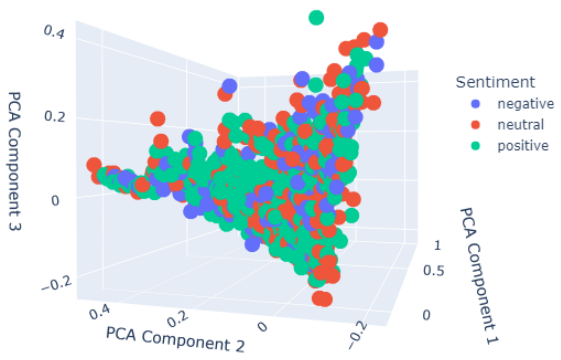

Figure 1. The threedimensional visualization of the Principle Component Analysis of the vectorized news sentiment shows a similar distribution of data throughout the three classes.

3.2.2 Random Forest

The Random Forest (RF) classifier operates by constructing multiple decision trees on various subsets of the data and then aggregating their predictions. Its ensemble architecture mitigates overfitting and can efficiently capture non-linear relationships. By processing the numerical vectors from TF-IDF or Word2Vec as input features, the RF identifies patterns or thresholds in the vector dimensions that are most indicative of sentiment. This makes it particularly effective for sentiment analysis, as it can leverage both the semantic relationships from embeddings and the relevance of specific terms highlighted by TF-IDF, while remaining robust to noise and irrelevant features.

3.2.3 Support Vector Machine

The Support Vector Machine (SVM), similar to the RF Classifier, is a powerful tool for sentiment analysis, though it approaches the task differently. While the RF leverages an ensemble of decision trees to capture non-linear relationships, the SVM relies on identifying a hyperplane that optimally separates data points into different classes. Its kernel-based framework enables the SVM to map complex textual data into higher-dimensional spaces, making it particularly effective for handling nuanced sentiment patterns. In this paper the radial base function is used. By utilizing numerical vectors from TF-IDF or Word2Vec, both classifiers excel at extracting meaningful features, with the RF offering robustness to noise and the SVM providing precision in boundary determination.

3.2.4 Bidirectional Encoder Representations from Transformers (BERT)

BERT is a transformer-based model designed to understand the context of words in a sequence by leveraging bidirectional attention. Unlike earlier models that processed text in a unidirectional manner, BERT captures context from both the left and right sides of a token, resulting in a deeper understanding of language. At its core, BERT employs a multi-head self-attention mechanism within a deep Transformer architecture, allowing it to focus on relevant parts of the input sequence while maintaining global context. During pre-training, BERT uses two objectives: Masked Language Modeling, where a portion of tokens are masked and the model predicts them based on their context, and Next Sentence Prediction, which helps understand relationships between sentences. These pre-training objectives equip BERT with a robust understanding of language semantics, making it adaptable to various downstream tasks.

FinBERT [5] builds on this foundational architecture but specializes in financial sentiment analysis by further adapting BERT to the financial domain. Because FinBERT undergoes additional pre-training on a domain-specific financial corpus (e.g., 'TRC2-financial') it's able to capture the specialized language and context of financial texts, such as jargon, abbreviations, and domain-specific semantics, which are crucial for accurately classifying sentiment in financial news, reports, and announcements. By focusing on sentiment-indicative terms in financial texts and leveraging domain-specific insights, FinBERT achieves state-of-the-art performance, offering a robust tool for analyzing the nuanced sentiments that drive financial decision-making. Unlike classic BERT, FinBERT's tailored training process eliminates the need for extensive external preprocessing, making it a streamlined and effective solution for financial sentiment analysis.

3.2.5 Large Language Model

The Lama3.1:latest model incorporates a large-scale language architecture with eight billion parameters, relying on multi-head attention and extensive training to capture diverse linguistic patterns. While theoretically well suited for text classification, its size imposes substantial computational demands. The observed performance limitations highlight the importance of model efficiency and suggest that, for sentiment-focused tasks, a carefully selected yet compact Transformer can be more advantageous than overlarge variants. The data was prompted to the model with the task for classifying the sentiment.

3.3 Application Architecture

The application is divided into the two components visualized in Figure 2.

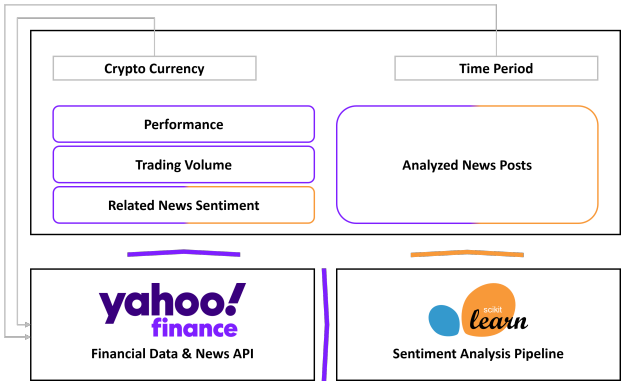

Figure 2. The applications frontend takes a cryptocurrency and time period input, fetches the relevant financial data and news articles from Yahoo Finance through their API, displays the financial data and news, while the pipeline classifies the news' sentiment.

The first one is the frontend delivering the visual representation of a cryptocurrencies performance and trading volume as well as the time-related sentiment analysis. The frontend offers the possibility for choosing the desired cryptocurrency as well as an individual period of time. The currencies performance in that time frame is displayed with a candle diagram, while the traded volume can be found in a bar chart below the performance graph. The time-related sentiment is displayed below the financial information using a stacked bar chart. For deeper analysis of the sentiment, the analized text instance are labeled and displayed and the right side of the frontend.

This data is retrieved and created by the backend consisting of corresponding API calls to Yahoo Finance for both the financial data as well as the last 200 crypto related news. While the financial data is simply displayed, the news are delivered as input for a sklearn pipeline including the preprocessing steps and the model prediction.

4 Dataset

The necessary data for the supervised training, testing, and cross-validating of the models requires two features. These are the textual posts themselves and their associated sentiment labels consisting of positive, neutral and negative. Although there are openly available datasets of such kind [6] [7], their labeling was mostly carried out by AI models, which results in dramatic losses in its quality. Additionally, the models trained on such datasets tend to fitting to the structure with which the ai model labeled the dataset rather than the sentiment itself, which then results in the same restricted quality with which the dataset was labeled in the first place.

Therefore, a manual revision of 2500 instance of the predefined datasets [6] labels was conducted to increase its accuracy. This data was then balanced and split 80/20 into train and test datasets. A second, already manually labeled and reviewed dataset [8] was used for cross-validating the models to ensure their generalizability.

5 Evaluation

The evaluation is conducted by using our manually labeled dataset as training and test dataset which is then cross-validated with the manually labeled dataset from IEEE. The metric used is the macro-avaraged F1 score which evaluates the balance between precision and recall across all classes, treating each class equally regardless of its frequency in the dataset. Although we use balanced datasets, the score is still fair, as it evaluates how well the model performs across all classes without skewing toward a specific class. This is espacially usefull in evaluating the trinary classification which produces nine possible classes. We will always prefer higher evaluation metrics on the cross-validation dataset since this proofs our model to be more generalizable and therefore more effective in our application.

Table 1 shows the results for trinary classification for the both simple classification algorithms for every combination of vectorization and preprocessing. The left value is always the evaluation metric on the training dataset and the right one on the cross-validation dataset. The SVM performes the best out of all model configurations in the table using Word2Vec as well as the group A of the preprocessing pipeline.

It can generally be observed, that group B of the preprocessing pipeline always performes worse with all models while A seems to often be the best solution closly followed by C. This gives a hint at the fact, that the models perform best when having access to the most tokens without removing named entities.

Categories		**RDF**	**SVM**
Vectorizer	**Prep**		
TF-IDF	A	0.81 / 0.34	0.82 / 0.40
	B	0.81 / 0.34	0.83 / 0.37
	C	0.81 / 0.36	0.83 / 0.41
Tri-Grams	A	0.78 / 0.35	0.82 / 0.40
	B	0.79 / 0.32	0.83 / 0.37
	C	0.81 / 0.35	0.85 / 0.42
Word2Vec	A	0.79 / 0.44	**0.71 / 0.49**
	B	0.79 / 0.38	0.69 / 0.47
	C	0.81 / 0.41	0.71 / 0.48

Table 1. Trinary classification for all combinations of classifiers and preprocessing methods with the SVM on Word2Vec and preprocessing pipeline A performing the best. A = compound word splitting + code sequence removal + NLTK tokenizing and stopword removal, B = compound word splitting + code sequence removal + ticker replacement + URL removal, NLTK tokenizing and stopword removal, C = compound word splitting + code sequence removal + URL removal + NLTK tokenizing and stopword removal + Lemmatizing and POS tagging.

Table 2 representing the binary classifications shows pretty much the same phenomens except the SVM now performing better with the preprocessing Group B by a tiny margin. This could be tight to the little amount of data on which the models are trained resulting in such a small but still odd outbreak.

Categories		**RDF**	**SVM**
Vectorizer	**Prep**		
TF-IDF	A	0.82 / 0.51	0.90 / 0.58
	B	0.83 / 0.49	0.90 / 0.55
	C	0.86 / 0.56	0.89 / 0.57
Tri-Grams	A	0.82 / 0.51	0.91 / 0.58
	B	0.83 / 0.48	0.89 / 0.54
	C	0.83 / 0.55	0.90 / 0.59
Word2Vec	A	0.86 / 0.61	0.83 / 0.63
	B	0.89 / 0.56	**0.82 / 0.65**
	C	0.86 / 0.59	0.83 / 0.64

Table 2. Binary classification for all combinations of classifiers and preprocessing methods with the SVM on Word2Vec and preprocessing pipeline B performing the best. A = compound word splitting + code sequence removal + NLTK tokenizing and stopword removal, B = compound word splitting + code sequence removal + ticker replacement + URL removal, NLTK tokenizing and stopword removal, C = compound word splitting + code sequence removal + URL removal + NLTK tokenizing and stopword removal + Lemmatizing and POS tagging.

Table 4 shows the results for our pretrained deep-learning models which perform ths best out of the models used for trinary classifiaction with OLLAMA reaching 0.51 and FinBERT scoring 0.55. This is understandable

since these models are way more complex and required a lot more data and parameter to be trained compared to our classificators. The models accuracy was increased by evaluating them on our own manually labeled dataset. With FinBERT performing the best it was our choise to be used in the active learning approach.

Modell	Accuracy
OLLAMA	0.51
FinBERT	**0.55**

Table 3. Comparsion of the accuracy of the pre-trained deep learning models with FinBERT outscoring OLLAMA with an accuracy of 0.55 on trinary classification.

Table 4 shows the results for the SVM trained on the active learning dataset produced by the FinBERT model. The choise for the SVM being trained on the active leraning dataset resulted in it being constantly the most precise in the previous evaluations. In the trinary classification it achives the greatest performance yet with an macro-avaraged F1 score of 0.50 for the cross-validation dataset which in this case is our manually labeled dataset and not the labeled IEEE dataset because of the underlying similarity in the data with the dataset labeled by the FinBERT model in our active learning approach. This increase compared to the 0.49 resulting from the model trained on our manually labeled dataset could be a result of the grater amount of data, although the increase in performance is only very small. Therefore further analysis on the improvement for binary classification wasn't conducted.

Categories		SVM
Vectorizer	**Prep**	
TF-IDF	A	0.86 / 0.42
	B	0.86 / 0.41
	C	0.85 / 0.44
Tri-Grams	A	0.86 / 0.42
	B	0.86 / 0.41
	C	0.86 / 0.44
Word2Vec	A	0.79 / 0.44
	B	0.73 / 0.50
	C	**0.75 / 0.50**

Table 4. Evaluation of the Support Vector Machine trained on the active learning data set labeled by the pre-trained FinBERT model for trinary classification performing best on Word2Vec and preprocessing pipeline C

Compared to the results on our first datasets in Appendix A which have been labeled by other ai models the manually labeled and the ones labeled through active learning show major improvements proving the bad quality of the openly available datasets.

6 Discussion

We faced the greatest challenge with the very basis of training ai models being the data itself. As a result of the unavailability of reliable and labeled datasets in the crypto domain we were left with the task of manual revision. Though a second manually labeled dataset was to be found and active learning proved to be a viable method, the limited instances restricted us in the choise of models, since deep learning models would only show their potential with a drastically increased size of high quality data sets.

Additionally, the size of the data sets didn't allow our models to be really excel at the task of reliably discriminating positive and negative sentiment from neutral sentiment limiting the correct visual representation of the peoples current crypto related sentiment in our application.

7 Conclusion

This work emphasizes the importance of high-quality, manually labeled data for effective sentiment analysis in the cryptocurrency domain. While such datasets outperform automated alternatives, they come with challenges of limited size and reduced vocabulary. Contrary to expectations, extensive data cleansing, such as removing named entities, reduced model performance, suggesting that context-specific terms influence sentiment classification. Active learning with FinBERT proved effective for trinary classification, though binary classification yielded the greatest accuracy overall. These findings highlight the need for tailored methods and balanced approaches to data quality and quantity, especially in dynamic fields like cryptocurrency sentiment analysis.

A Appendix

Categories		RDF	SVM
Vectorizer	**Prep**		
TF-IDF	A	0.81 / 0.34	**0.82 / 0.40**
	B	0.81 / 0.34	0.83 / 0.37
	C	0.81 / 0.36	0.83 / 0.41
Tri-Grams	A	0.86 / 0.32	**0.82 / 0.40**
	B	0.85 / 0.33	0.83 / 0.37
	C	0.83 / 0.35	0.85 / 0.42
Word2Vec	A	0.73 / 0.36	0.66 / 0.37
	B	0.73/ 0.37	0.64 / 0.38
	C	0.74/ 0.36	0.65 / 0.38

Table 5. Trinary classification for all combinations of classifiers and preprocessing methods with the SVM on TF-IDF/Tri-Grams and preprocessing pipeline A trained on [6] and cross-validated on [7] performing the best. A = compound word splitting + code sequence removal + NLTK tokenizing and stopword removal, B = compound word splitting + code sequence removal + ticker replacement + URL removal, NLTK tokenizing and stopword removal, C = compound word splitting + code sequence removal + URL removal + NLTK tokenizing and stopword removal + Lemmatizing and POS tagging.

Categories		RDF	SVM
Vectorizer	**Prep**		
TF-IDF	A	0.89 / 0.53	0.89 / 0.54
	B	0.89 / 0.52	0.89 / 0.54
	C	0.88 / 0.54	0.89 / 0.55
Tri-Grams	A	0.88 / 0.53	0.88 / 0.54
	B	0.89 / 0.53	0.87 / 0.58
	C	0.88 / 0.54	0.88 / 0.56
Word2Vec	A	0.80 / 0.61	0.77 / 0.67
	B	0.81/ 0.63	**0.75 / 0.68**
	C	0.80/ 0.62	0.76 / 0.67

Table 6. Binary classification for all combinations of classifiers and preprocessing methods with the SVM on TF-IDF/Tri-Grams and preprocessing pipeline A trained on [6] and cross-validated on [7] performing the best. A = compound word splitting + code sequence removal + NLTK tokenizing and stopword removal, B = compound word splitting + code sequence removal + ticker replacement + URL removal, NLTK tokenizing and stopword removal, C = compound word splitting + code sequence removal + URL removal + NLTK tokenizing and stopword removal + Lemmatizing and POS tagging.

Modell	Accuracy
OLLAMA	0.39
FinBERT	**0.41**

Table 7. Comparsion of the accuracy of the pre-trained deep learning models evaluated on the data set [6] with FinBERT outscoring OLLAMA with an accuracy of 0.41 on trinary classification.

References

[1] K.I. Roumeliotis, N.D. Tselikas, D.K. Nasiopoulos, Big Data and Cognitive Computing **8**, 63 (2024)

[2] R.N. Satrya, O.N. Pratiwi, R.Y. Farifah, J. Abawajy, *Cryptocurrency Sentiment Analysis on the Twitter Platform Using Support Vector Machine (SVM) Algorithm*, in *2022 International Conference Advancement in Data Science, E-learning and Information Systems (ICADEIS)* (IEEE, 2022), pp. 01–05, ISBN 978-1-66546-387-4, https://ieeexplore.ieee.org/document/10037413/

[3] N. Aslam, F. Rustam, E. Lee, P.B. Washington, I. Ashraf, Sentiment Analysis and Emotion Detection on Cryptocurrency Related Tweets Using Ensemble LSTM-GRU Model **10**, 39313 (2022)

[4] KA-KA-shi, *Googlenews-vectors-negative300 (word2vec)* (n.a), https://www.kaggle.com/datasets/adarshsng/googlenewsvectors/data

[5] D. Araci, *Finbert: Financial sentiment analysis with pre-trained language models* (2019), 1908.10063, https://arxiv.org/abs/1908.10063

[6] Oliviervha, *Crypto news* + (2024), https://www.kaggle.com/datasets/oliviervha/crypto-news

[7] K. Dave, *Indiafinancesent corpus* (2024), [https://huggingface.co/datasets/kdave/Indian_Financial_News]

[8] B. Taborda, A. de Almeida, J. Carlos Dias, F. Batista, R. Ribeiro, *Stock market tweets data* (2021), https://dx.doi.org/10.21227/g8vy-5w61