Modellvergleich und Optimierung von NLP-Verfahren zur Klassifikation kryptobezogener Reddit-Beiträge

Digital Business University of Applied Sciences

Data Science & Business Analytics
DMI01 Data Mining
Daniel Ambach

Eingereicht von Dennis Reimer
Martrikelnummer: 190288
Datum: 16.04.2025

Zusammenfassung

Diese Arbeit befasst sich mit dem Vergleich und der Optimierung von NLP-Modellen zur Sentimentanalyse von Reddit-Beiträgen im Kontext von Kryptowährungen. Dabei werden sowohl klassische, lexikonbasierte Verfahren als auch moderne, transformerbasierte Sprachmodelle untersucht. Ziel ist es, diese Modelle im Hinblick auf ihre Fähigkeit zu bewerten, Aussagen auf Reddit zuverlässig in die Sentimentklassen *bullish*, *neutral* oder *bearish* einzuordnen. Besonderes Augenmerk liegt auf der Differenzierung zwischen vollständigen Reddit-Posts und kurzen Kommentaren, da sich diese in Struktur, Sprache und Tonalität deutlich unterscheiden.

Die Untersuchung basiert auf einem kombinierten Datensatz aus etwa 750 manuell gelabelten Reddit-Posts sowie einem öffentlich verfügbaren Kommentar-Datensatz aus einer IEEE-Studie, in dem das Sentiment direkt von den Beitragenden angegeben wurde. Die methodische Umsetzung folgt dem CRISP-DM-Prozessmodell, das sich durch eine klare Strukturierung aller Projektphasen auszeichnet. Zum Einsatz kommen unter anderem FinBERT und CryptoBERT als spezialisierte transformerbasierte Modelle, während VADER und TextBlob als klassische Vergleichsverfahren dienen.

Die Ergebnisse zeigen, dass FinBERT auf den längeren Reddit-Posts nur moderate Verbesserungen durch das Fine-Tuning erzielen konnte, während CryptoBERT auf den Kommentaren nach gezielter Modellanpassung eine sehr hohe Klassifikationsgüte erreichte (F1-Score: 0.92). Damit unterstreicht die Arbeit die Bedeutung domänenspezifischer Modellierung und einer sorgfältigen Hyperparameteroptimierung für den erfolgreichen Einsatz von Sentimentanalysen im Bereich von Social Media und Kryptowährungen.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Analyse von Meinungen und Emotionen im digitalen Raum hat in den letzten Jahren stark an Bedeutung gewonnen, besonders im Umfeld sozialer Medien und Finanzmärkte. Plattformen wie Reddit fungieren dabei als wichtige Diskussionsorte, an denen sich private und institutionelle Anlegerinnen und Anleger über Kryptowährungen, Markttrends und Anlagestrategien austauschen. Die automatisierte Auswertung solcher Inhalte mit Hilfe von Verfahren der Sentimentanalyse kann wertvolle Hinweise auf kollektive Stimmungen liefern und dadurch zu fundierteren Entscheidungen beitragen.

Eine besondere Herausforderung bei Reddit besteht in der Vielfalt und Unstrukturiertheit der Texte. Die Inhalte reichen von ausführlichen Diskussionen bis hin zu kurzen, sarkastischen Kommentaren oder humorvollen Memes. Klassische, lexikonbasierte Ansätze stoßen hier schnell an ihre Grenzen, da ihnen das notwendige Kontextverständnis fehlt. Transformerbasierte Sprachmodelle wie BERT, FinBERT oder CryptoBERT gehen über eine reine Wortebene hinaus und berücksichtigen die Bedeutung eines Satzes im Gesamtzusammenhang. Gerade bei komplexen oder indirekt formulierten Aussagen ist dies ein entscheidender Vorteil (Huyen, 2022, Kapitel 4–5).

Diese Arbeit verfolgt das Ziel, verschiedene NLP-Modelle zur Sentimentanalyse von Reddit-Beiträgen mit Bezug zu Kryptowährungen zu vergleichen und im nächsten Schritt gezielt zu optimieren. Untersucht werden sowohl transformerbasierte Modelle wie FinBERT, CryptoBERT, RoBERTa und DeBERTa als auch klassische Ansätze wie VADER und TextBlob. Die Modelle werden dabei auf zwei unterschiedlichen Textarten getestet: auf vollständigen Reddit-Posts und auf kurzen Kommentaren. Anschließend werden die leistungsfähigsten Modelle weiter angepasst und bewertet.

Die methodische Umsetzung folgt dem CRISP-DM-Prozessmodell (Cross Industry Standard Process for Data Mining), das sich als Standard zur Strukturierung datengetriebener Projekte etabliert hat (Bramer, 2007, S. 15–16). Alle sechs Phasen – von der Zieldefinition bis zur abschließenden Bewertung – werden im Rahmen dieser Arbeit dokumentiert und umgesetzt.

Für die Analyse wurden zwei Datensätze herangezogen: Zum einen eine eigene Sammlung von etwa 750 manuell gelabelten Reddit-Posts, zum anderen ein frei zugänglicher Kommentar-Datensatz aus einer IEEE-Studie, bei dem das Sentiment direkt von den Verfassenden angegeben wurde (IEEE, 2023). Die technische Umsetzung erfolgte in Python unter Verwendung der Bibliotheken transformers, datasets und scikit-learn. Das Fine-Tuning der Modelle wurde auf GPU-beschleunigten Systemen durchgeführt (Downey & Lang, 2024, Kapitel 2).

Die Arbeit ist in fünf Kapitel gegliedert. Kapitel 2 stellt die theoretischen Grundlagen vor, insbesondere zu Sentimentanalyse, Modellarten und Metriken. Kapitel 3 beschreibt das methodische Vorgehen entlang des CRISP-DM-Prozesses. Kapitel 4 enthält die empirischen Ergebnisse und deren Auswertung. Kapitel 5 fasst die zentralen Erkenntnisse zusammen und gibt einen Ausblick auf weiterführende Fragestellungen.

2 Theoretischer Hintergrund

Um die eingesetzten Modelle und Bewertungsmethoden einordnen zu können, werden in diesem Kapitel die Grundlagen der Sentimentanalyse, relevante Modellarten und Bewertungsmetriken sowie das zugrunde liegende methodische Vorgehen vorgestellt.

2.1 Grundlagen der Sentimentanalyse

Sentimentanalyse ist ein Teilbereich des Natural Language Processing (NLP) und befasst sich mit der automatisierten Erkennung und Klassifikation von Meinungsäußerungen in Texten. Sie zielt darauf ab, Inhalte etwa als *positiv*, *negativ* oder *neutral* zu bewerten. Ihre Anwendungen reichen von Produktbewertungen über politische Stimmungsanalysen bis hin zur Finanzmarktforschung.

Besonders in sozialen Medien wie Reddit stellt die Analyse eine Herausforderung dar: Die Texte sind oft kurz, informell, enthalten Ironie, Emojis oder Bezugnahmen auf vorherige Kontexte. Klassische regelbasierte Verfahren scheitern häufig an diesen Phänomenen, da ihnen das nötige Kontextverständnis fehlt. Deshalb kommen zunehmend tiefe Sprachmodelle auf Basis von Transformer-Architekturen zum Einsatz, die semantische Zusammenhänge modellieren können (Huyen, 2022, Kapitel 4).

2.2 Klassifikationsansätze im Vergleich

Grundsätzlich lassen sich Sentimentklassifikatoren in zwei Gruppen unterteilen: lexikonbasierte Verfahren und transformerbasierte Modelle.

Die Lexikonbasierte Verfahren beinhalten Modelle wie VADER (Valence Aware Dictionary and sEntiment Reasoner) und TextBlob und verwenden vorab definierte Wortlisten, in denen Begriffen Polarisierungswerte zugewiesen sind. Die Gesamtwertung eines Textes ergibt sich durch Aggregation dieser Einzelwerte. VADER wurde speziell für Social Media konzipiert und berücksichtigt auch Großschreibung, Interpunktion und Emoticons. TextBlob verwendet ein einfaches regelbasiertes Verfahren in Kombination mit statistischen Textmerkmalen.

Der Vorteil dieser Methoden liegt in ihrer schnellen Einsatzfähigkeit und hohen Interpretierbarkeit. Ihre Schwäche besteht darin, dass sie keine semantischen Zusammenhänge erkennen – ein wesentlicher Nachteil bei komplexer Sprache oder impliziten Meinungen.

Transformer sind eine moderne Architektur innerhalb künstlicher neuronaler Netze. Sie basieren auf dem Self-Attention-Mechanismus, der es ermöglicht, die Bedeutung eines Wortes im Kontext aller anderen Wörter zu erfassen. Dadurch sind sie besonders leistungsfähig bei der Analyse semantisch komplexer Texte (Huyen, 2022, Kapitel 4).

Besonders relevant für diese Arbeit sind:

FinBERT: Ein auf Finanzsprache spezialisiertes Modell, das auf Wirtschafts- und Börsentexten trainiert wurde.

CryptoBERT: Eine Variante, die auf Krypto-Diskussionen in Foren und sozialen Medien feinjustiert wurde.

RoBERTa und DeBERTa: Architekturverbesserungen von BERT, die auf allgemeinen Textdaten trainiert wurden und ohne Domänenspezialisierung arbeiten.

Transformerbasierte Modelle sind rechenintensiv und benötigen größere Datenmengen, bieten jedoch ein deutlich tieferes Sprachverständnis – ein entscheidender Vorteil für Reddit-Daten mit ihrer hohen stilistischen Vielfalt.

2.3 Data Mining und CRISP-DM-Prozess

Die in dieser Arbeit durchgeführten Schritte lassen sich vollständig in das CRISP-DM-Modell (Cross Industry Standard Process for Data Mining) einordnen – ein weit verbreitetes Framework zur Strukturierung datengetriebener Projekte (Bramer, 2007, S. 15–16). Es besteht aus sechs klar definierten Phasen:

1. Business Understanding: Problemdefinition, Zielsetzung
2. Data Understanding: Beschreibung, Exploration und Bewertung der Datenqualität
3. Data Preparation: Bereinigung, Vorverarbeitung, Labeling und Formatierung
4. Modeling: Auswahl geeigneter Modelle und Konfigurationsparameter
5. Evaluation: Modellvergleich, Interpretation der Metriken
6. Deployment: Anwendung oder Transfer der Ergebnisse (in dieser Arbeit nur theoretisch)

Der Vorteil von CRISP-DM liegt in der Trennung von Analyse und Umsetzung, wodurch sowohl technische Details als auch fachliche Zielsetzungen klar nachvollziehbar bleiben (Bramer, 2007, S. 15-16). In dieser Arbeit bildet CRISP-DM die methodische Klammer, an der sich alle Kapitel orientieren.

2.4 Bewertung von Klassifikationsmodellen

Die Bewertung der Modellleistung erfolgt typischerweise über folgende Metriken:

Accuracy: Anteil korrekt klassifizierter Instanzen

F1-Score (macro): Harmonie von Precision und Recall über alle Klassen hinweg, sinnvoll bei unbalancierten Daten

Cohen's Kappa: Maß für die Übereinstimmung zwischen Vorhersage und Ground Truth, bereinigt um Zufallstreffer

Besonders bei unausgeglichenen Datensätzen – wie bei Reddit-Kommentaren – ist der F1-Score (macro) die zuverlässigere Metrik. Er bewertet die Leistung jeder Klasse gleich und lässt sich daher gut für Modellvergleiche einsetzen (Huyen, 2022, Kapitel 10).

3 Implementierung und Methodik

Im Folgenden wird das methodische Vorgehen der Arbeit detailliert beschrieben. Die Struktur orientiert sich vollständig am CRISP-DM-Prozessmodell (Bramer, 2007, S. 15–16), das eine systematische Umsetzung datengetriebener Projekte ermöglicht.

3.1 Datenaufbereitung und Split

Nach dem Import der Datensätze wurden diese bereinigt, vereinheitlicht und in numerische Labels für die drei Sentimentklassen *bullish* (0), *neutral* (1) und *bearish* (2) überführt. Die Daten wurden anschließend in Trainings-, Validierungs- und Testdaten unterteilt. Dabei wurde explizit darauf geachtet, dass die Klassenverteilung auch in den Teilmengen erhalten bleibt (stratifizierter Split).

Die konkrete Aufteilung und das Verhältnis (z. B. 60 % Training, 20 % Validierung, 20 % Test) variierten je nach Datensatz leicht und wurden direkt im Notebook durchgeführt. Auf einen externen Split-Helfer wurde bewusst verzichtet, um die Kontrolle über die Aufteilung direkt im Hauptnotebook zu behalten.

3.2 Baseline- Vergleich der Modelle

Zur Evaluation wurden zunächst sechs verschiedene Modelle auf einem festen Testset verglichen. Die Modellklasse reichte von lexikonbasierten Verfahren (VADER, TextBlob) bis zu transformerbasierten Klassifikatoren (FinBERT, CryptoBERT, RoBERTa, DeBERTa). Die lexikonbasierten Modelle erhielten eine explizite Schwellenwertlogik zur Klassenzuweisung:

```
score = model_obj.polarity_scores(text)["compound"]
preds.append(0 if score >= 0.05 else 2 if score <= -0.05 else 1)
```

Abbildung 1: Klassifizierung eines Textes in bullish, neutral und bearish basierend auf dem compound-Score von Vader

Nach der Inferenz wurde für jedes Modell ein Klassifikationsreport generiert. Die F1-Scores der drei Klassen wurden extrahiert und zentral gespeichert:

```python
def extract_f1_scores(classification_reports):
    f1_data = {}
    for model, report in classification_reports.items():
        f1_data[model] = {
            "bullish": report["bullish"]["f1-score"],
            "neutral": report["neutral"]["f1-score"],
            "bearish": report["bearish"]["f1-score"]
        }
    return f1_data
```

Abbildung 2: Funktion zur Extraktion der F1-Scores pro Klasse aus einem gespeicherten Klassifikationsreport

Die aufbereiteten Scores wurden später zur Erstellung der Balkendiagramme in Kapitel 4 verwendet.

3.3 Modelloptimierung mit Custom- Trainer

Für die zwei besten Modelle – FinBERT (Posts) und CryptoBERT (Kommentare) – wurde ein gezieltes Fine-Tuning durchgeführt. Dabei kam ein Custom-Trainer mit gewichteter Verlustfunktion zum Einsatz, um die unausgeglichene Klassenverteilung zu berücksichtigen:

```python
class WeightedLossTrainer(Trainer):
    def __init__(self, *args, loss_fn=None, **kwargs):
        super().__init__(*args, **kwargs)
        self.loss_fn = loss_fn

    def compute_loss(self, model, inputs, return_outputs=False, **kwargs):
        labels = inputs.pop("labels")
        outputs = model(**inputs)
        logits = outputs.logits
        loss = self.loss_fn(logits, labels)
        return (loss, outputs) if return_outputs else loss
```

Abbildung 3: Implementierung eines gewichteten Custom- Trainers zur Berücksichtigung unbalacierter Klassen

Es wurde mit unterschiedlichen Gewichtungen experimentiert, bis das beste Ergebnis gespeichert wurde. Die technische Umsetzung erfolgte mithilfe der Trainer-API von Hugging Face und der transformers-Bibliothek. Die gewählte Architektur erlaubte es, benutzerdefinierte Verlustfunktionen (z. B. CrossEntropyLoss mit Gewichtung) direkt in das Trainingssetup zu integrieren (Huyen, 2022, Kapitel 6).

3.4 Vergleich verschiedener Lernrraten

Ein zentrales Element der Optimierung war ein systematischer Vergleich unterschiedlicher Lernraten (2e-5 bis 1e-6). Für jede Rate wurde ein vollständiger Trainingslauf durchgeführt und die F1-Makro-Scores pro Epoche mitgeloggt.

Die Trainingsargumente wurden so gesetzt, dass nach jeder Epoche eine Validierung erfolgte. Zusätzlich wurde ein Early-Stopping-Mechanismus integriert, um Überanpassung zu vermeiden.

```
results = {}
lrs = [2e-5, 1e-5, 5e-6, 3e-6, 1e-6]

for lr in lrs:
    print(f"Starte Training für learning_rate = {lr}")

    output_dir = os.path.join(MODEL_PATHS["finbert_posts"], f"finetuned_lr_{lr}")
    logging_dir = os.path.join("../../logs/finbert_posts", f"lr_{lr}")
    os.makedirs(output_dir, exist_ok=True)
    os.makedirs(logging_dir, exist_ok=True)

    training_args = TrainingArguments(
        output_dir=output_dir,
        evaluation_strategy="epoch",
        save_strategy="epoch",
        learning_rate=lr,
        per_device_train_batch_size=8,
        per_device_eval_batch_size=8,
        num_train_epochs=10,
        weight_decay=0.01,
        load_best_model_at_end=True,
        metric_for_best_model="f1",
        logging_dir=logging_dir,
        logging_strategy="epoch",
        report_to="none",
        remove_unused_columns=False,
        disable_tqdm=True
    )
```

Abbildung 4:Trainingsschleife zum Vergleich verschiedener Lernraten mittels Trainingarguments in Hugging Face

Die daraus entstandenen Lernkurven sind in Kapitel 4 abgebildet und wurden zur Auswahl des besten Modells verwendet. Die Sensitivität der Modelle gegenüber Lernraten war besonders bei kleinen Datensätzen deutlich zu beobachten, was ein bekanntes Phänomen bei modernen NLP-Architekturen ist (Huyen, 2022, Kapitel 7).

4 Ergebnisse und Diskussion

Ziel dieses Abschnitts ist die systematische Analyse und Bewertung der getesteten Modelle, dabei wird zwischen den beiden Datentypen- Posts und Kommentare unterschieden. Zusätzlich werden die Effekte des Fine-Tunings auf die Modellperformance untersucht.

4.1 Modellvergleich auf untrainierten Baselines

Abbildung 5 und 6 zeigen die F1-Scores der getesteten Modelle auf dem ursprünglichen, untrainierten Stand. Dabei wurde jeweils ein gemeinsames Testset verwendet. Auffällig ist, dass kein Modell alle drei Klassen konsistent gut erkennt.

Bei Posts (Abb. 5) schnitten VADER und TextBlob insgesamt stabil ab, insbesondere bei *bearish*. Transformer-Modelle wie DeBERTa und RoBERTa hatten starke Einbrüche bei *bearish* (F1 teils < 0.1).

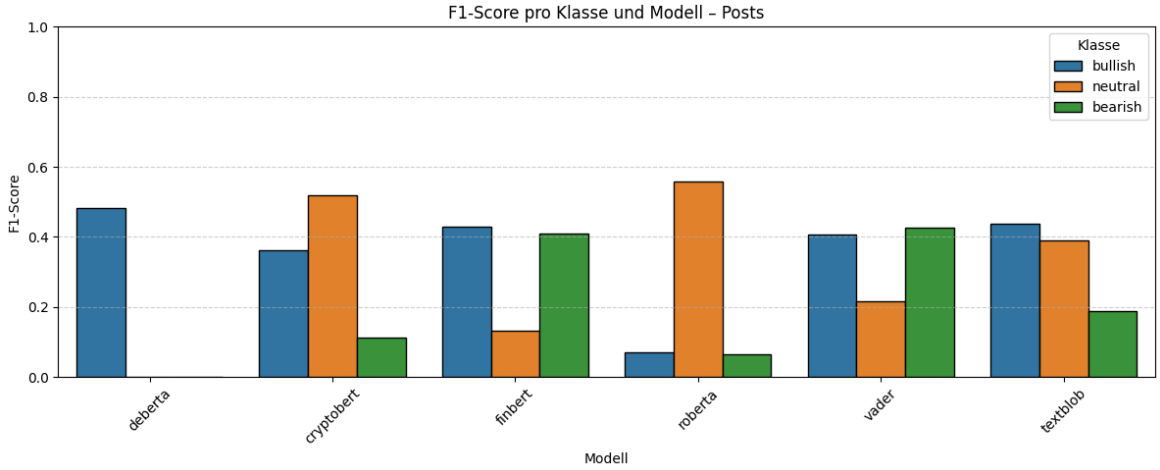

Abbildung 5: F1-Scores der Ausgsangsmodelle auf Reddit-Posts

Bei Kommentaren (Abb. 6) zeigte CryptoBERT auch ohne Fine-Tuning bereits eine sehr starke Performance (F1 bis zu 0.91 für *bullish*), während DeBERTa hier erneut nahezu vollständig versagte.

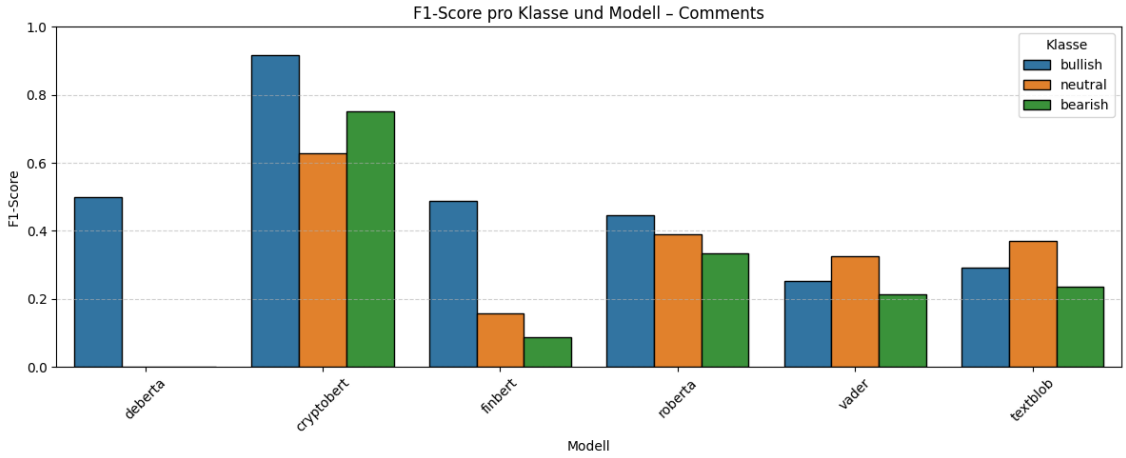

Abbildung 6: F1-Scores der Ausgangsmodelle auf Reddit-Kommentaren

Diese Ergebnisse verdeutlichen zwei zentrale Punkte:

1. Die lexikonbasierten Modelle liefern bei kurzen Texten (*Kommentare*) keine belastbaren Ergebnisse.
2. Transformer-Modelle brauchen kontext- und domänenspezifisches Fine-Tuning, um ihr Potenzial auszuschöpfen.

4.2 Fine-Tuning Ergebnisse

Nach gezieltem Fine-Tuning auf jeweils den geeignetsten Modellen wurden deutliche Verbesserungen erzielt.

CryptoBERT wurde auf dem externen Datensatz mit 999 Kommentaren feinjustiert. Die Ergebnisse nach zehn Epochen und optimierter Lernrate (5e-6) sind in Tabelle 1 zusammengefasst:

- F1-Score (macro): 0.9199
- Cohen's Kappa: 0.8814
- Sehr hohe Scores in allen drei Klassen: bullish (0.96), neutral (0.88), bearish (0.92)

Diese Ergebnisse übertreffen alle anderen Modelle deutlich. Besonders positiv fällt auf, dass auch *bearish* zuverlässig erkannt wird – was ohne Fine-Tuning nicht gelang.

FinBERT wurde auf einem eigenen Datensatz mit 298 Reddit-Posts trainiert. Trotz gezieltem Feintuning und Lernratenvergleich blieb die Modellgüte deutlich hinter CryptoBERT zurück:

- F1-Score (macro): 0.5329
- Cohen's Kappa: 0.2895
- Klassenunterschiede: *bullish* (0.55), *neutral* (0.62), *bearish* (0.43)

Die moderate Performance kann auf die begrenzte Datenmenge (~750 Posts) zurückgeführt werden. Zudem enthalten Posts oft komplexere Satzstrukturen und weniger klare Stimmungsindikatoren als Kommentare.

4.3 Lernratenvergleich

Abbildung 3 und 4 zeigen die Lernkurven des F1-Scores (macro) auf dem Validierungsset für verschiedene Lernraten:

Bei CryptoBERT (Abb. 7) zeigten sich klare Unterschiede. Die besten Ergebnisse wurden bei 5e-6 und 3e-6 erzielt, die auch langfristig stabil blieben. Höhere Raten (1e-5, 2e-5) führten zu instabilen Verläufen.

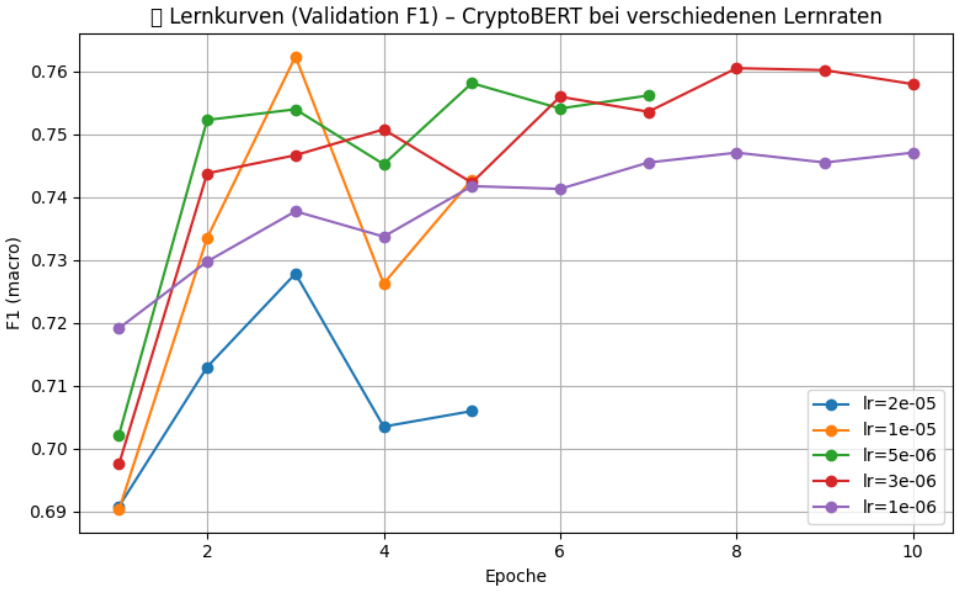

Abbildung 7: Lernkurven(F1 macro) für CryptoBERT bei verschiedenen Lernratenauf dem Kommentar-Datensatz

Bei FinBERT (Abb. 8) war die Lernkurve weniger klar ausgeprägt. Zwar zeigten 3e-6 und 5e-6 ebenfalls stabile Resultate, jedoch ohne signifikante Trennung zu den schlechteren Raten.

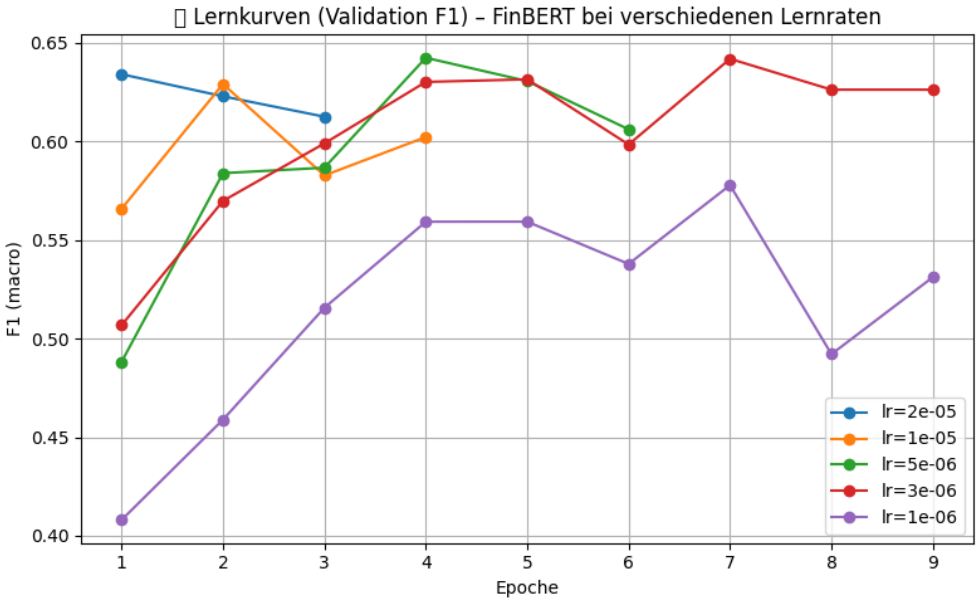

Abbildung 8: Lernkurven (F1 macro) für FinBERT bei verschiedenen Lernraten auf dem Post-Datensatz

Diese Beobachtungen bestätigen die hohe **Hyperparameter-Sensitivität** von Transformer-Modellen bei kleinen Datensätzen – ein bekanntes Phänomen, das in der Literatur regelmäßig diskutiert wird (Huyen, 2022, Kapitel 7).

4.4 Gesamtbewertung

Die besten Ergebnisse wurden mit CryptoBERT nach Fine-Tuning erzielt. Die sehr hohe Klassifikationsgüte bei Kommentaren (F1 > 0.91) zeigt, dass domänenspezifisches Vorwissen und Fine-Tuning auf strukturell passenden Daten entscheidend sind. FinBERT hingegen blieb hinter den Erwartungen zurück – vermutlich primär aufgrund der limitierten Datenbasis.

Die Ergebnisse legen nahe, dass in zukünftigen Projekten der Fokus verstärkt auf:

- skalierbare Datenbeschaffung (mehr manuelle Labels oder automatisiertes Bootstrapping),
- und auf datengetriebene Hyperparameteroptimierung gelegt werden sollte.

5 Fazit

Ziel dieser Arbeit war es, verschiedene NLP-Modelle zur Sentimentanalyse von Reddit-Beiträgen rund um das Thema Kryptowährungen systematisch zu untersuchen. Dabei wurden sowohl klassische, lexikonbasierte Verfahren als auch moderne, transformerbasierte Sprachmodelle auf ihre Leistungsfähigkeit hin verglichen und anschließend im Rahmen eines gezielten Fine-Tunings weiter optimiert. Im Mittelpunkt stand die Frage, inwiefern sich

Aussagen nicht nur thematisch, sondern auch hinsichtlich ihrer emotionalen Tonalität zuverlässig den Kategorien *bullish*, *neutral* und *bearish* zuordnen lassen. Gerade in informellen, ironischen oder sehr kurzen Kommentaren, wie sie auf Reddit häufig vorkommen, stellt diese Aufgabe eine besondere Herausforderung dar.

Die Ergebnisse zeigen klar, dass transformerbasierte Modelle den klassischen Ansätzen in fast allen Bewertungsdimensionen überlegen sind, sofern sie auf ausreichend domänenspezifischem Material trainiert und mit passenden Hyperparametern angepasst werden. Besonders überzeugend war die Leistung von CryptoBERT. Nach dem Fine-Tuning auf dem Kommentar-Datensatz erreichte das Modell einen macro-F1-Score von 0.92 und einen Cohen's Kappa von 0.88. Damit konnte es nicht nur alle drei Sentimentklassen präzise unterscheiden, sondern auch durchweg stabile Vorhersagen liefern. Das ist ein deutliches Zeichen für seine Robustheit im Vergleich zu den übrigen Modellen.

FinBERT hingegen, das auf einem kleineren Datensatz gelabelter Reddit-Posts trainiert wurde, zeigte durch das Fine-Tuning nur begrenzte Verbesserungen. Zwar ließ sich auch hier ein Anstieg der Modellgüte erkennen, insbesondere in der *bearish*-Klasse, doch blieb das Gesamtergebnis hinter den Erwartungen zurück. Vermutlich liegt das vor allem an der geringen Datenmenge sowie an der sprachlich komplexeren Struktur von Reddit-Posts, die eine eindeutige Sentimentzuordnung erschweren. Auch ergänzende Maßnahmen wie Data Augmentation führten in diesem Fall nicht zu einer nennenswerten Verbesserung. Das deutet darauf hin, dass transformerbasierte Modelle wie FinBERT stark auf inhaltlich vielfältige und realistische Trainingsdaten angewiesen sind.

5.1 Ausblick

Die hier vorgestellten Ergebnisse bieten eine solide Grundlage für zukünftige Arbeiten, in denen transformerbasierte Modelle gezielt für Social-Media-Analysen im Finanzbereich eingesetzt werden sollen. Ein entscheidender Hebel zur Verbesserung der Modellleistung liegt vermutlich in der skalierbaren und thematisch passenden Erweiterung der Trainingsdaten. Techniken wie semi-supervised Learning, Active Learning oder Pseudo-Labeling könnten dabei helfen, unannotierte Reddit-Beiträge effizient in das Trainingsverfahren einzubeziehen und so die Datenbasis deutlich zu verbreitern (Huyen, 2022, Kapitel 10).

Darüber hinaus wäre es sinnvoll, die in dieser Arbeit erzielten Ergebnisse durch den Einsatz erklärbarer KI-Methoden wie LIME oder SHAP zu ergänzen. Solche Verfahren könnten dazu beitragen, die Entscheidungslogik der Modelle – insbesondere bei knappen, mehrdeutigen oder widersprüchlichen Sentiment-Signalen – besser nachvollziehbar zu machen. Das würde

nicht nur die Transparenz erhöhen, sondern auch das Vertrauen in die Modellvorhersagen stärken.

Eine weitere vielversprechende Perspektive wäre die Integration der trainierten Modelle in ein interaktives Dashboard oder eine Webanwendung. Damit könnten aktuelle Stimmungen auf Reddit automatisch ausgewertet und visuell dargestellt werden. In Kombination mit Marktdaten ließe sich ein solches System als praktisches Werkzeug für Analystinnen und Analysten, Investorinnen und Investoren oder auch Forschende einsetzen.

Zusammenfassend lässt sich sagen, dass transformerbasierte Sentimentmodelle bei passender Modellarchitektur, sauberer Datenbasis und gut abgestimmten Parametern einen bedeutenden Beitrag zur automatisierten Stimmungsanalyse in digitalen Diskursräumen leisten können. Auch wenn Herausforderungen wie Ironie, unstrukturierte Sprache und begrenzte Datenmengen weiterhin bestehen, bieten moderne Verfahren bereits heute vielversprechende Ansätze für eine differenzierte und zuverlässige Analyse kollektiver Meinungen.

Literaturverzeichnis

Bramer, M. (2007). *Principles of data mining* (Vol. 180). London: Springer.

Downey, A. B., & Lang, J. W. (2024). *Python lernen mit KI-Tools: Einstieg in die Programmierung mit KI-Unterstützung*. o'Reilly Verlag GmbH & Co. KG.

Huyen, C. (2022). *Designing machine learning systems: An iterative process for production-ready applications*. O'Reilly Media.

IEEE (2023). *Sentiment analysis of crypto-related Reddit posts using transformer models*.