



Hochschule für angewandte Wissenschaften Coburg

Fakultät Elektrotechnik und Informatik

Studiengang: Informatik

Bachelorarbeit

Multi-Modal Feature Fusion with Cross-Attention for Tabular and Textual Data

Städler, Sebastian

Abgabe der Arbeit: 16.12.2025

Betreut durch:

Prof. Dr. Roman Rischke, Hochschule Coburg

Inhaltsverzeichnis

1 Einführung	1
2 Theoretische Grundlagen	2
2.1 Multimodale Daten	2
2.1.1 Heterogenität in multimodalen Daten	2
2.1.2 Repräsentation von Textdaten	2
2.1.3 Repräsentation tabellarischer Daten	4
2.2 Transformer-Architekturen und Attention-Mechanismen	6
2.2.1 Der Transformer-Encoder	6
2.2.2 Scaled Dot-Product Attention	7
2.2.3 Self-Attention vs. Cross-Attention	7
2.3 Strategien der multimodalen Fusion	9
2.3.1 Mechanismen der Fusion	9
2.3.2 Architekturen und Positionierung der Fusion	10
3 Verwandte Arbeiten	12
4 Methodik	13
5 Ergebnisse	14
6 Diskussion und Fazit	15
Literaturverzeichnis	II
Ehrenwörtliche Erklärung	VI

1 Einführung

Das ist meine Einfuehrung ...

2 Theoretische Grundlagen

2.1 Multimodale Daten

2.1.1 Heterogenität in multimodalen Daten

Multimodale Daten, wie Texte und Tabellen, stellen eine Herausforderung dar, da jede Modalität eine eigene Struktur, Semantik und statistische Verteilung besitzt [?]. Textdaten sind sequenziell und linguistisch strukturiert, wobei Informationen über Tokens verteilt sind und Semantik aus Kontext und Reihenfolge entsteht. Im Gegensatz dazu sind tabellarische Daten attributbasiert und nicht-sequenziell; jedes Feature trägt eine explizite Bedeutung, und die Spaltenreihenfolge ist irrelevant.

Diese strukturellen Unterschiede führen zu einem „Manifold Mismatch“ [?, HKCK20]. Text wird in hochdimensionalen, kontinuierlichen Embedding-Räumen repräsentiert, während Tabulardaten aus heterogenen, unskalierten Merkmalen bestehen. Die geometrische Inkompatibilität dieser Räume verhindert, dass Attention-Mechanismen, die auf Text-Tokens effektiv angewendet werden können, direkt auf rohe Tabularfeatures funktionieren. Dies liegt daran, dass Tabularfeatures weder normalisiert sind noch eine Tokenstruktur oder semantische Distanz aufweisen.

Für eine sinnvolle Fusion von Text und Tabular in multimodalen Modellen ist daher eine Projektion in einen gemeinsamen Embedding-Space unerlässlich. Modelle müssen Mechanismen zur Harmonisierung der Skalen, zur Tokenisierung tabellarischer Merkmale und zur Erzeugung eines gemeinsamen semantischen Raums bereitstellen. Die Entwicklung solcher Mechanismen ist ein zentrales Thema in der Forschung zu multimodalen Modellen.

2.1.2 Repräsentation von Textdaten

Für die effektive Verarbeitung natürlicher Sprache auch genannt Natural Language Processing (NLP) und die in dieser Arbeit untersuchte Fusion von Text- und Tabellendaten ist die Transformation textueller Informationen in numerische Vektorrepräsentationen unerlässlich [AU22].

Zentrale Begriffe sind dabei:

- **Text Vektorisierung:** Umwandlung von Text in numerische Vektoren, notwendig für maschinelle Lernalgorithmen [AU22].
- **Embedding:** Gelernte, dichte und niedrigdimensionale Vektorrepräsentation von Text (Wörtern, Sätzen etc.), die semantische Beziehungen abbildet (z. B. „king – man + woman ≈ queen“) [MCCD13].
- **Repräsentation:** Jede Form der Textkodierung im Modell.

Historisch betrachtet basierten frühe Ansätze auf frequenzbasierten Verfahren wie dem Bag-of-Words-Modell oder der Term Frequency-Inverse Document Frequency (TF-IDF) [SB88]. Diese Methoden repräsentieren Dokumente als Vektoren von Worthäufigkeiten, ignorieren jedoch weitgehend die grammatischen Struktur und die semantische Bedeutung der Wörter. Zudem führen sie oft zu hochdimensionalen, dünn besetzten Vektoren (sparse vectors).

Einen signifikanten Fortschritt markierte die Einführung von verteilten Wortrepräsentationen (Word Embeddings), insbesondere durch das Word2Vec-Verfahren [MCCD13]. Hierbei werden Wörter in einen dichten, niedrigdimensionalen Vektorraum projiziert, wobei diese Projektionen durch einfache neuronale Netze gelernt werden. Semantisch ähnliche Wörter liegen in diesem Vektorraum nahe beieinander. Ein wesentlicher Nachteil dieser statischen Embeddings besteht jedoch darin, dass jedem Wort unabhängig von seinem Kontext ein fixer Vektor zugewiesen wird. Polyseme Wörter, die je nach Satzkontext unterschiedliche Bedeutungen haben, können so nicht adäquat abgebildet werden.

Um dieses Defizit zu beheben, wurden kontextuelle Embeddings entwickelt. Peters et al. stellten mit Embeddings from Language Models (ELMo) einen Ansatz vor, der tiefere neuronale Netze nutzt, um kontextabhängige Wortvektoren zu generieren [PNI⁺18]. Den aktuellen Standard setzen jedoch Transformer-basierte Modelle wie Bidirectional Encoder Representations from Transformers (BERT), eingeführt von Devlin et al. [DCLT19]. Als Encoder-only-Modell wird BERT auf großen Textmengen vor-trainiert und erzeugt durch ein bidirektionales Training tiefgreifende kontextuelle Repräsentationen. Im Gegensatz zu ELMo basiert BERT auf der Transformer-Architektur und nutzt intensiv Self-Attention-Mechanismen, deren technische Details in Abschnitt 2.2 erläutert werden.

Für die Verarbeitung ganzer Sätze oder Dokumente, wie sie in dieser Arbeit relevant ist, stoßen reine Token-Embeddings jedoch an Grenzen. Hier kommen Techniken wie Pooling oder die Verwendung spezieller Tokens ins Spiel. Insbesondere das `[CLS]`-Token (Classifier Token) in Modellen wie BERT wird häufig genutzt, um eine aggregierte Repräsentation des gesamten Satzes zu erhalten. Dieses spezielle Token wird jeder Eingabesequenz vorangestellt und sammelt während der Verarbeitung durch die Transformer-Schichten über Self-Attention kontextuelle Informationen des gesamten Inputs. Der finale Vektor des `[CLS]`-Tokens ($T_{[CLS]}$) dient somit als kompakte Repräsentation der gesamten Eingabe und wird in Klassifikationsaufgaben üblicherweise verwendet, um die Vorhersage zu treffen [DCLT19]. Während Modifikationen wie Sentence-BERT (SBERT) [RG19] darauf abzielen, semantisch aussagekräftige Satz-Embeddings zu erzeugen, liegt der Fokus dieser Arbeit auf der direkten Nutzung von `[CLS]`-Tokens für die Satzrepräsentation. Diese stabilen semantischen Kodierungen auf Satzebene schaffen die notwendige Voraussetzung, um Textinformationen effizient mit anderen Modalitäten, wie tabellarischen Merkmalen, in einer multimodalen Architektur zu fusionieren.

2.1.3 Repräsentation tabellarischer Daten

Während Textdaten, wie im vorangegangenen Kapitel beschrieben, eine sequentielle Struktur aufweisen, unterscheiden sich tabellarische Daten fundamental in ihrer Beschaffenheit. Sie bilden die Grundlage zahlreicher Anwendungen in der Industrie, stellen jedoch für Deep-Learning-Ansätze eine besondere Herausforderung dar [BLS⁺24]. Um eine effektive Fusion mit Textmodalitäten in einer Transformer-Architektur zu ermöglichen, ist eine Transformation der rohen Tabellendaten in eine kompatible Repräsentation notwendig.

Heterogenität und Permutationsinvarianz Im Gegensatz zu homogenen Datenquellen wie Bildern (Pixelwerte) oder Texten (Wort-Tokens) zeichnen sich tabellarische Daten durch eine starke Heterogenität aus. Sie bestehen aus einem Mix von dichten numerischen Features (kontinuierliche Werte wie Preis oder Alter) und sparsen kategorialen Features (diskrete Werte wie Postleitzahl oder Produktkategorie) [BLS⁺24, BSP23].

Ein weiteres wesentliches Merkmal ist die Permutationsinvarianz der Spalten. Anders als Wörter in einem Satz, deren Position die semantische Bedeutung maßgeblich bestimmt (vgl. „Hund beißt Mann“ vs. „Mann beißt Hund“), haben die Spalten einer Tabelle keine natürliche Ordnung. Ein Modell muss daher robust gegenüber der Vertauschung von Feature-Positionen sein, solange die Zuordnung von Wert und Feature-Typ erhalten bleibt [GRKB21]. Die Repräsentation ($Feature_A, Feature_B$) muss für das Netzwerk semantisch identisch zu ($Feature_B, Feature_A$) verarbeitet werden.

Grenzen klassischer Vorverarbeitungsmethoden In traditionellen Machine-Learning-Verfahren, wie Gradient Boosted Decision Trees (GBDT), werden kategoriale Daten häufig mittels One-Hot-Encoding (OHE) verarbeitet. Dabei wird eine kategoriale Variable mit der Kardinalität C in einen binären Vektor der Länge C transformiert. Für Deep-Learning-Modelle und insbesondere für die Integration in Transformer-Architekturen weist dieses Verfahren jedoch gravierende Nachteile auf:

Zum einen führt OHE bei Features mit hoher Kardinalität zu extrem hochdimensionalen und dünn besetzten Vektoren (Sparsity). Dies erschwert es neuronalen Netzen, dichte und aussagekräftige Repräsentationen zu lernen, da der Großteil der Eingabewerte Null ist [HKCK20, GRKB21]. Zum anderen fehlt OHE jegliche semantische Information. Alle Kategorien werden als orthogonal betrachtet, was bedeutet, dass der Abstand zwischen allen Kategorien im Vektorraum identisch ist. Semantische Beziehungen, wie etwa die Ähnlichkeit zwischen den Kategorien „Auto“ und „LKW“ im Vergleich zu „Apfel“, können durch OHE nicht abgebildet werden [HKCK20].

Auch einfache Multi-Layer Perceptrons (MLPs) stoßen auf rohen Daten an ihre Grenzen, da sie numerische Eingaben lediglich als skalare Werte betrachten und im ersten Schritt keine feature-

spezifische Verarbeitung ermöglichen, wie sie beispielsweise für das Erlernen von Interaktionen zwischen spezifischen Features notwendig wäre [GRKB21].

Feature Tokenization und Latenter Raum Um die Inkompatibilität zwischen den heterogenen Tabellendaten und den textbasierten Transformer-Modellen zu überwinden, ist eine Projektion der Features in einen gemeinsamen latenten Raum erforderlich. Dieser Prozess wird in der Literatur als *Feature Tokenization* bezeichnet [BLS⁺24, BSP23].

Das Ziel ist es, eine Tabellenzeile x in eine Sequenz von Vektoren \mathbf{E} zu transformieren, die strukturell der Eingabe eines Sprachmodells gleicht:

$$\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_N] \quad \text{mit} \quad \mathbf{e}_i \in \mathbb{R}^H \quad (2.1)$$

Hierbei entspricht H der Dimension des Hidden-States des verwendeten Transformer-Modells (z. B. $H = 768$ bei BERT). Durch diese Angleichung der Dimensionen wird die technische Voraussetzung geschaffen, um später Mechanismen wie Cross-Attention zwischen Text- und Tabellen-Tokens anzuwenden [BSP23]. Im Gegensatz zu OHE erlaubt dieser dichte Vektorraum das Erlernen semantischer Ähnlichkeiten, sodass verwandte Features im Vektorraum geometrisch näher beieinander liegen.

Methoden der Transformation Für die Erzeugung dieser Token-Sequenz haben sich verschiedene Ansätze etabliert, die sich insbesondere in der Behandlung numerischer Werte unterscheiden.

Für **kategoriale Features** wird analog zu Word Embeddings im NLP eine Lookup-Tabelle gelernt. Jede Kategorie wird auf einen trainierbaren Vektor projiziert, der während des Trainings optimiert wird, um den semantischen Kontext der Kategorie zu erfassen [HKCK20].

Für **numerische Features** ist eine Lookup-Tabelle nicht direkt anwendbar, da die Werte kontinuierlich sind. Hier kommen spezialisierte Verfahren zum Einsatz:

- *Lineare Projektion*: Der skalare Wert wird direkt mittels einer linearen Schicht in den hochdimensionalen Raum \mathbb{R}^H projiziert.
- *Binning/Quantisierung*: Der Wertebereich wird in diskrete Intervalle unterteilt, die dann wie kategoriale Werte behandelt werden können. Dies ermöglicht dem Modell, nicht-lineare Zusammenhänge besser zu erfassen [SGS⁺21, GRKB21].

Architekturen der Repräsentation Hinsichtlich der Struktur der erzeugten Sequenz lassen sich zwei Hauptstrategien unterscheiden, die in aktuellen Forschungsarbeiten vorgeschlagen werden [BSP23]:

Der erste Ansatz, bekannt als **Column-wise Embeddings** (Spaltenweise Einbettung), bildet jedes Feature der Ursprungstabelle auf genau einen Token-Vektor ab. Dies wird beispielsweise im *TabTransformer* [HKCK20] oder im *SAINT*-Modell [SGS⁺21] angewendet. Die Sequenzlänge N entspricht hierbei der Anzahl der Spalten. Der Vorteil liegt in der Erhaltung der feingranularen Information jedes einzelnen Features, was die Interpretierbarkeit durch Attention-Weights erleichtert.

Der zweite Ansatz verfolgt eine **globale Projektion** (Latente Tokenization). Hierbei werden alle Features einer Zeile zunächst konateniert und durch ein MLP geleitet, um eine Sequenz von latenten Tokens zu erzeugen, die nicht mehr zwangsläufig 1-zu-1 den ursprünglichen Spalten entsprechen [GB21]. Dies ermöglicht eine Entkopplung der Sequenzlänge von der Anzahl der Eingabefeatures und kann die Rechenkomplexität in der nachfolgenden Attention-Schicht reduzieren. Zudem erhält das Modell so die Möglichkeit, bereits in der Tokenization-Phase globale Zusammenhänge zwischen den Features zu aggregieren.

Unabhängig von der gewählten Methode ist das Ergebnis dieser Transformation eine Sequenz dichter Vektoren. Erst durch diesen Schritt wird die „Sprache“ der Tabelle in die „Sprache“ des Transformers übersetzt, was die Basis für die in dieser Arbeit untersuchte multimodale Fusion mittels Cross-Attention bildet.

2.2 Transformer-Architekturen und Attention-Mechanismen

Nachdem im vorangegangenen Kapitel die Grundlagen der Datenrepräsentation für Text und Tabellen erläutert wurden, widmet sich dieser Abschnitt der Modellarchitektur, die den aktuellen Stand der Technik in der Verarbeitung natürlicher Sprache definiert: dem Transformer. Ursprünglich für maschinelle Übersetzungsaufgaben konzipiert, hat sich diese Architektur als extrem leistungsfähig für das Erlernen komplexer Zusammenhänge in sequenziellen Daten erwiesen. Das Verständnis der internen Mechanismen, insbesondere der Attention, ist essenziell für die in dieser Arbeit entwickelte multimodale Fusion.

2.2.1 Der Transformer-Encoder

Das Transformer-Modell wurde 2017 von Vaswani et al. eingeführt und markierte einen Paradigmenwechsel im Deep Learning, da es vollständig auf Rekurrenz und Faltung verzichtet und stattdessen ausschließlich auf Attention-Mechanismen setzt [VSP⁺17]. Die ursprüngliche Architektur besteht aus zwei Hauptkomponenten: einem Encoder, der die Eingabesequenz verarbeitet und in eine abstrakte Repräsentation überführt, und einem Decoder, der basierend darauf eine Ausgabesequenz generiert.

Für die Aufgabenstellung dieser Arbeit, bei der es um die Klassifikation von multimodalen Daten geht, ist primär der Encoder-Teil von Relevanz. Encoder-only-Modelle, wie das in Abschnitt 2.1.2 schon vorgestellte BERT, nutzen einen Stapel von Transformer-Blöcken, um für jedes Eingabe-Token einen kontextualisierten Vektor zu berechnen [DCLT19]. Im Gegensatz zu statischen Embeddings (siehe Abschnitt 2.1.2), bei denen ein Wort stets denselben Vektor erhält, fließen im Transformer-Encoder Informationen aus dem gesamten Kontext der Sequenz in die Repräsentation jedes einzelnen Tokens ein. Der Encoder fungiert somit als eine hochkomplexe „Verstehens-Maschine“, die syntaktische und semantische Beziehungen innerhalb der Daten extrahiert.

2.2.2 Scaled Dot-Product Attention

Das Herzstück eines jeden Transformer-Blocks ist der Attention-Mechanismus. Er ermöglicht es dem Modell, dynamisch zu entscheiden, welche Teile der Eingabe für den aktuellen Verarbeitungsschritt relevant sind. Vaswani et al. formalisieren dies als *Scaled Dot-Product Attention* [VSP⁺17].

Das Konzept lässt sich durch eine Analogie aus dem Information Retrieval beschreiben: Eine Abfrage (*Query Q*) wird gegen eine Menge von Schlüsseln (*Keys K*) abgeglichen, um die passenden Werte (*Values V*) zu extrahieren. Mathematisch wird dies durch folgende Gleichung ausgedrückt:

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (2.2)$$

Hierbei berechnet das Skalarprodukt QK^T die Ähnlichkeit zwischen der Query und allen Keys. Der Faktor $\frac{1}{\sqrt{d_k}}$ dient der Skalierung, um zu verhindern, dass die Werte für die Softmax-Funktion zu groß werden, was zu verschwindenden Gradienten führen würde [VSP⁺17]. Die Softmax-Funktion normalisiert diese Ähnlichkeitswerte zu Gewichten, die sich zu 1 aufsummieren. Schließlich wird eine gewichtete Summe der Values V gebildet. Intuitiv bedeutet dies: Ein Vektor Q „sucht“ in den Vektoren K nach relevanten Informationen und zieht sich die entsprechenden Inhalte aus V .

2.2.3 Self-Attention vs. Cross-Attention

Ein entscheidender Aspekt für das Verständnis moderner Transformer-Architekturen – und insbesondere für multimodale Modelle – ist die Herkunft der Vektoren Q , K und V . Hierbei wird zwischen *Self-Attention* und *Cross-Attention* unterschieden.

Self-Attention (Intra-Modal) Im Standard-Encoder, wie er von Vaswani et al. [VSP⁺17] beschrieben wird, stammen Query, Key und Value aus derselben Quelle (z. B. der Einbettung des vorherigen Layers). Dies wird als Self-Attention bezeichnet.

- **Funktionsweise:** Jedes Token der Sequenz betrachtet jedes andere Token derselben Sequenz, um seinen eigenen Kontext zu schärfen.
- **Zweck:** Aufbau eines tiefen Verständnisses innerhalb einer Modalität. So kann das Modell beispielsweise auflösen, worauf sich ein Pronomen in einem Satz bezieht oder wie die Bedeutung eines Wortes durch seine Nachbarn beeinflusst wird.

Cross-Attention (Inter-Modal) Für die Fusion unterschiedlicher Modalitäten, wie Text und Tabellendaten, ist die Cross-Attention von zentraler Bedeutung. Dieses Konzept findet sich prominent in multimodalen Erweiterungen wie LXMERT [TB19].

- **Funktionsweise:** Die Vektoren stammen aus unterschiedlichen Quellen. Beispielsweise generiert Modalität A (z. B. Text) die Queries Q , während Modalität B (z. B. Tabelle) die Keys K und Values V liefert.
- **Zweck:** Dies ermöglicht den direkten Informationsfluss über Modalitätsgrenzen hinweg. Die Text-Tokens können gezielt Informationen aus den Tabellen-Features „abfragen“, die für den aktuellen Kontext relevant sind. Dies bildet das theoretische Fundament für die in dieser Arbeit entwickelten Fusions-Architekturen, da es eine flexible und dynamische Verknüpfung der heterogenen Datenräume erlaubt.

2.3 Strategien der multimodalen Fusion

Multimodale Fusion bezeichnet den Prozess der Integration von Informationen aus verschiedenen Modalitäten, um eine gemeinsame Vorhersage (beispielsweise Klassifikation oder Regression) zu berechnen [LT24, BAM19]. Das Ziel ist es, die Stärken der einzelnen Datenquellen zu kombinieren und so eine leistungsfähigere Repräsentation zu schaffen, als es mit einer einzelnen Modalität möglich wäre.

Die Notwendigkeit, Daten aus unterschiedlichen Quellen zu fusionieren, ergibt sich aus mehreren fundamentalen Vorteilen [LT24, BAM19]. Erstens bieten verschiedene Modalitäten oft komplementäre Informationen. Sie beobachten denselben Sachverhalt aus unterschiedlichen Perspektiven, wodurch Aspekte sichtbar werden, die in einer isolierten Betrachtung verborgen blieben [LT24, BAM19]. Zweitens erhöht die Fusion die Robustheit des Systems. Sollte eine Modalität verdeckt sein oder aufgrund von Sensorausfällen fehlen, können die verbleibenden Modalitäten die fehlenden Informationen kompensieren, was die Fehleranfälligkeit gegenüber unimodalen Ansätzen deutlich reduziert [BAM19]. Ein kritischer Erfolgsfaktor ist dabei das sogenannte *Alignment*. Damit die Fusion sinnvoll arbeitet, müssen die Modalitäten semantisch korrekt aufeinander abgebildet werden, etwa durch eine Projektion in einen gemeinsamen Repräsentationsraum. Ohne ein solches Alignment können falsche Zuordnungen entstehen, wodurch die fusionierte Darstellung inkonsistent wird [JGF⁺24, LT24].

2.3.1 Mechanismen der Fusion

Die technische Realisierung der Fusion, also *wie* die Vektoren der unterschiedlichen Modalitäten zusammengeführt werden, kann durch verschiedene mathematische Operationen erfolgen [JGF⁺24, BAM19]. Ein zentrales Konzept ist hierbei die *Joint Representation*. Das Ziel ist es, die unimodalen Eingangsvektoren in einen gemeinsamen semantischen Unterraum zu projizieren, um eine einzige, multimodale Repräsentation Z zu erzeugen [JGF⁺24, LT24]. Mathematisch lässt sich dieser Vorgang allgemein formulieren als:

$$Z = f(X_{\text{text}}, X_{\text{tab}}) \quad (2.3)$$

Wobei f eine Funktion darstellt (beispielsweise ein neuronales Netz oder eine deterministische Operation), die aus den unimodalen Repräsentationen X_{text} und X_{tab} die fusionierte multimodale Repräsentation Z berechnet [BAM19].

Zu den gängigsten spezifischen Operationen zählen:

Concatenation (Verkettung) Dies ist die wohl einfachste Form der Fusion, bei der die Merkmalsvektoren lediglich hintereinander gehängt werden:

$$Z = [X_{\text{text}}, X_{\text{tab}}] \quad (2.4)$$

Ein wesentlicher Vorteil ist, dass die Eingangsvektoren X_{text} und X_{tab} nicht dieselbe Dimension besitzen müssen. Die Dimension des Ergebnisvektors Z entspricht der Summe der Einzeldimensionen. In der Literatur wird dieser Ansatz oft als „Early Fusion“ bezeichnet, wenn er direkt auf der Ebene der Eingangsmerkmale angewendet wird [BAM19].

Addition (Summierung) Hierbei werden die Vektoren elementweise addiert:

$$Z = X_{\text{text}} + X_{\text{tab}} \quad (2.5)$$

Dies setzt zwingend voraus, dass beide Vektoren dieselbe Dimension aufweisen, weshalb oft eine vorherige Projektion notwendig ist. Diese Methode ist besonders in Joint-Representation-Ansätzen verbreitet [JGF⁺24].

Attention-Mechanismen Im Gegensatz zu statischen Operationen wie der Addition oder Verkettung erlauben Attention-Mechanismen eine dynamische Gewichtung der Features. Besonders relevant ist die *Cross-Attention*. Hierbei können Repräsentationen einer Zielmodalität (z. B. Text) gezielt Informationen aus einer Quellmodalität (z. B. Tabelle) abfragen. Typischerweise stammen dabei die *Queries* aus der Zielmodalität, während *Keys* und *Values* aus der Quellmodalität geliefert werden. Dies ermöglicht eine gewichtete Aggregation von Informationen, die auf den aktuellen Kontext konditioniert ist, und erlaubt die Modellierung komplexer Abhängigkeiten über Modalitätsgrenzen hinweg [LT24].

2.3.2 Architekturen und Positionierung der Fusion

Neben der Art der Operation ist entscheidend, *an welcher Stelle* im Modellfluss die Fusion stattfindet. Traditionell wird hierbei zwischen Early, Late und Hybrid Fusion unterschieden, wobei moderne Deep-Learning-Architekturen diese Grenzen zunehmend aufweichen [JGF⁺24, LT24, BAM19].

Early Fusion (Daten- und Feature-Ebene) Bei der Early Fusion findet die Integration unmittelbar nach der Extraktion der Merkmale oder sogar schon auf Ebene der Rohdaten statt, bevor diese die tieferen Schichten des Modells durchlaufen [JGF⁺24, LT24, BAM19]. Oft wird ein gemeinsamer Modellzweig (One-Tower-Architektur) verwendet, um die kombinierten Eingaben

gemeinsam zu verarbeiten [LT24]. Der Vorteil dieses Ansatzes liegt darin, dass Korrelationen und Interaktionen zwischen den „Low-Level“-Merkmalsen sehr früh gelernt werden können [BAM19]. Nachteilig ist jedoch die hohe Anforderung an die Synchronisation der Daten. Bei sehr heterogenen Modalitäten kann eine gemeinsame frühe Verarbeitung die Modellierung erschweren [JGF⁺24].

Late Fusion (Entscheidungs- und Ausgabe-Ebene) Hier erfolgt die Integration erst am Ende des Modells. Jede Modalität wird zunächst in einem eigenen Zweig (Two-Tower-Architektur) verarbeitet [LT24]. Die resultierenden High-Level-Repräsentationen werden anschließend fusioniert, beispielsweise durch ein Voting-Schema, Mittelwertbildung oder ein nachgeschaltetes MLP [BAM19]. Dies bietet hohe Flexibilität, da für jede Modalität spezialisierte Modelle verwendet werden können, und erleichtert den Umgang mit fehlenden Daten [BAM19]. Allerdings werden bei der Late Fusion potenzielle Interaktionen auf niedrigerer Ebene ignoriert. Fehlerhafte Daten in einer Modalität können das Endergebnis direkt negativ beeinflussen, da keine frühe Korrektur durch die andere Modalität möglich ist [BAM19].

Hybrid und Deep Fusion Hybride Ansätze versuchen, die Vorteile beider Welten zu kombinieren [JGF⁺24, BAM19]. Bei der *Deep Fusion* oder Feature-Level Fusion geschieht die Integration nicht nur am Anfang oder Ende, sondern auch in den Zwischenschichten des Modells [JGF⁺24]. Dies wird oft durch eine „Two-Leg“-Struktur realisiert, bei der eine zusätzliche Fusionskomponente auf zwei separaten Verarbeitungszweigen aufsetzt [LT24]. Moderne Transformer-basierte Ansätze treiben dies noch weiter: Durch Mechanismen wie Cross-Attention in jeder Schicht interagieren die Modalitäten kontinuierlich über die gesamte Tiefe des Modells hinweg [LT24]. Dies ermöglicht eine deutlich tiefere Integration als einfache Kombinationen.

3 Verwandte Arbeiten

4 Methodik

5 Ergebnisse

6 Diskussion und Fazit

Literaturverzeichnis

- [AU22] ABUBAKAR, H. D. ; UMAR, M.: Sentiment Classification: Review of Text Vectorization Methods: Bag of Words, Tf-Idf, Word2vec and Doc2vec. In: *SLUJST* 4 (2022), Aug, Nr. 1 & 2, S. 27–33. <http://dx.doi.org/10.56471/slujst.v4i.266>. – DOI 10.56471/slujst.v4i.266
- [BAM19] BALTRUSAITIS, T. ; AHUJA, C. ; MORENCY, L.-P.: Multimodal Machine Learning: A Survey and Taxonomy. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41 (2019), Feb, Nr. 2, S. 423–443. <http://dx.doi.org/10.1109/TPAMI.2018.2798607>. – DOI 10.1109/TPAMI.2018.2798607
- [BLS⁺²⁴] BORISOV, V. ; LEEMANN, T. ; SESSLER, K. ; HAUG, J. ; PAWELCZYK, M. ; KASNECI, G.: Deep Neural Networks and Tabular Data: A Survey. In: *IEEE Trans. Neural Netw. Learning Syst.* 35 (2024), Jun, Nr. 6, S. 7499–7519. <http://dx.doi.org/10.1109/TNNLS.2022.3229161>. – DOI 10.1109/TNNLS.2022.3229161
- [BSP23] BADARO, G. ; SAEED, M. ; PAPOTTI, P.: Transformers for Tabular Data Representation: A Survey of Models and Applications. In: *Transactions of the Association for Computational Linguistics* 11 (2023), S. 227–249. http://dx.doi.org/10.1162/tacl_a_00544. – DOI 10.1162/tacl_a_00544
- [DCLT19] DEVLIN, Jacob ; CHANG, Ming-Wei ; LEE, Kenton ; TOUTANOVA, Kristina: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019), May. <http://dx.doi.org/10.48550/arXiv.1810.04805>. – DOI 10.48550/arXiv.1810.04805
- [GB21] GU, K. ; BUDHKAR, A.: A Package for Learning on Tabular and Text Data with Transformers. In: ZADEH, A. (Hrsg.) ; MORENCY, L.-P. (Hrsg.) ; LIANG, P. P. (Hrsg.) ; ROSS, C. (Hrsg.) ; SALAKHUTDINOV, R. (Hrsg.) ; PORIA, S. (Hrsg.) ; CAMBRIA, E. (Hrsg.) ; SHI, K. (Hrsg.): *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*. Mexico City, Mexico : Association for Computational Linguistics, Jun 2021, S. 69–73
- [GRKB21] GORISHNIY, Y. ; RUBACHEV, I. ; KHRULKOV, V. ; BABENKO, A.: Revisiting Deep Learning Models for Tabular Data. In: *Advances in Neural Information Processing Systems* (2021)
- [HKCK20] HUANG, X. ; KHETAN, A. ; CVITKOVIC, M. ; KARNIN, Z.: TabTransformer: Tabular Data Modeling Using Contextual Embeddings. (2020), Dec. <http://dx.doi.org/10.48550/arXiv.2012.06678>. – DOI 10.48550/arXiv.2012.06678

- [JGF⁺24] JIAO, T. ; GUO, C. ; FENG, X. ; CHEN, Y. ; SONG, J.: A Comprehensive Survey on Deep Learning Multi-Modal Fusion: Methods, Technologies and Applications. In: *CMC-Computers, Materials & Continua* 80 (2024), Nr. 1, S. 1–35. <http://dx.doi.org/10.32604/cmc.2024.053204>. – DOI 10.32604/cmc.2024.053204
- [LT24] LI, S. ; TANG, H.: Multimodal Alignment and Fusion: A Survey. (2024). <http://dx.doi.org/10.48550/arXiv.2411.17040>. – DOI 10.48550/arXiv.2411.17040. – arXiv:2411.17040
- [MCCD13] MIKOLOV, Tomas ; CHEN, Kai ; CORRADO, Greg ; DEAN, Jeffrey: Efficient Estimation of Word Representations in Vector Space. (2013), Sep. <http://dx.doi.org/10.48550/arXiv.1301.3781>. – DOI 10.48550/arXiv.1301.3781
- [PNI⁺18] PETERS, Matthew E. ; NEUMANN, Mark ; IYYER, Mohit ; GARDNER, Matt ; CLARK, Christopher ; LEE, Kenton ; ZETTLEMOYER, Luke: Deep Contextualized Word Representations. In: WALKER, Marilyn (Hrsg.) ; JI, Heng (Hrsg.) ; STENT, Amanda (Hrsg.): *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana : Association for Computational Linguistics, Jun 2018, S. 2227–2237
- [RG19] REIMERS, Nils ; GUREVYCH, Iryna: Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. (2019), Aug. <http://dx.doi.org/10.48550/arXiv.1908.10084>. – DOI 10.48550/arXiv.1908.10084
- [SB88] SALTON, Gerard ; BUCKLEY, Chris: Term-weighting approaches in automatic text retrieval. In: *Information Processing & Management* 24 (1988), Nr. 5, S. 513–523. [http://dx.doi.org/10.1016/0306-4573\(88\)90021-0](http://dx.doi.org/10.1016/0306-4573(88)90021-0). – DOI 10.1016/0306-4573(88)90021-0
- [SGS⁺21] SOMEPALLI, G. ; GOLDBLUM, M. ; SCHWARZSCHILD, A. ; BRUSS, C. B. ; GOLDSTEIN, T.: SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training. In: *arXiv preprint arXiv:2106.01342* (2021). <https://arxiv.org/abs/2106.01342v1>
- [TB19] TAN, H. ; BANSAL, M.: LXMERT: Learning Cross-Modality Encoder Representations from Transformers. (2019), Dec. <http://dx.doi.org/10.48550/arXiv.1908.07490>. – DOI 10.48550/arXiv.1908.07490
- [VSP⁺17] VASWANI, Ashish ; SHAZER, Noam ; PARMAR, Niki ; USZKOREIT, Jakob ; JONES, Llion ; GOMEZ, Aidan N. ; KAISER, Łukasz ; POLOSUKHIN, Illia: Attention is

Literaturverzeichnis

All you Need. In: *Advances in Neural Information Processing Systems 30*, Curran Associates, Inc., 2017

Literaturverzeichnis

Persönliche Angaben / Personal details

Städler, Sebastian

Familienname, Vorname / Surnames, given names

19.03.2002

Geburtsdatum / Date of birth

Informatik

Studiengang / Course of study

00383022

Matrikelnummer / Student registration number

Eigenständigkeitserklärung

Declaration

Hiermit versichere ich, dass ich diese Arbeit selbstständig verfasst und noch nicht anderweitig für Prüfungszwecke vorgelegt habe. Ich habe keine anderen als die angegebenen Quellen oder Hilfsmittel benutzt. Die Arbeit wurde weder in Gänze noch in Teilen von einer Künstlichen Intelligenz (KI) erstellt, es sei denn, die zur Erstellung genutzte KI wurde von der zuständigen Prüfungskommission oder der bzw. dem zuständigen Prüfenden ausdrücklich zugelassen. Wörtliche oder sinngemäße Zitate habe ich als solche gekennzeichnet.

Es ist mir bekannt, dass im Rahmen der Beurteilung meiner Arbeit Plagiatserkennungssoftware zum Einsatz kommen kann.

Es ist mir bewusst, dass Verstöße gegen Prüfungsvorschriften zur Bewertung meiner Arbeit mit „nicht ausreichend“ und in schweren Fällen auch zum Verlust sämtlicher Wiederholungsversuche führen können.

I hereby certify that I have written this thesis independently and have not submitted it elsewhere for examination purposes. I have not used any sources or aids other than those indicated. The work has not been created in whole or in part by an artificial intelligence (AI), unless the AI used to create the work has been expressly approved by the responsible examination board or examiner. I have marked verbatim quotations or quotations in the spirit of the text as such.

I am aware that plagiarism detection software may be used in the assessment of my work.

I am aware that violations of examination regulations can lead to my work being graded as "unsatisfactory" and, in serious cases, to the loss of all repeat attempts.

Unterschrift Studierende/Studierender / Signature student

96450 Coburg, den 16.12.2025

Ort, Datum / Place, date