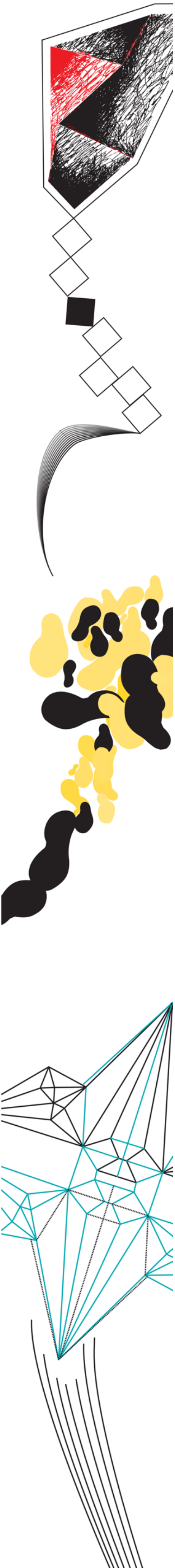


UNIVERSITY OF TWENTE.

MSc Thesis Computer Science

Co-Attention-Based Pairwise Learning for Author Name Disambiguation

Qiuke Li

August, 2022

Committee:	dr. Shenghui Wang dr. Doina Bucur Rob Koopman (OCLC B.V.)
Faculty:	Electrical Engineering, Mathematics and Computer Science (EEMCS)

Abstract

Digital library management systems suffer from inefficient retrieval caused by name ambiguity. Manual annotations require domain-specific knowledge and time-consuming cleaning work. Natural Language Processing and Deep neural networks are recently utilised to distinguish authorships of publications with identical author names. However, earlier machine learning approaches lack the latest embedding techniques in feature processing. Therefore, crucial latent information about record relationships is lost. Besides, no human-readable interpretation is provided. Based on state-of-art embedding techniques and attention mechanisms, this thesis proposed a co-attention-based pairwise learning model for author name disambiguation. The contribution of this thesis is threefold: first, it applies appropriate methods to process multiple types of features: textual, discrete, and coauthor features, with the goal of retaining all latent information of all components of records. Second, it engages the self-attention and co-attention mechanisms to investigate latent interactive information between records. Third, it provides explanations about model predictions by visualising the self-attention and co-attention weights. The experiment reveals that the co-attention-based model achieves the best scores using accuracy, F1, and ROC AUC measurements in most generated datasets. Although it is still debatable whether the attention weights are interpretations, they intuitively provide evidence of decision processes.

Acknowledgements

First and foremost, I would like to thank OCLC for giving me the opportunity to work on the author name disambiguation project, which gave me a solid understanding of the disambiguation process and deep collaboration within the company.

I would also like to express my deep gratitude to Rob Koopman for his patient guidance, enthusiastic encouragement and useful critiques of this research work. He has taught me the methodology to carry out the research as clearly as possible.

My sincere thanks also go to the university supervisors Dr. Shenghui Wang and Dr. Doina Bucur, who provided academic insight and expertise that greatly assisted the research and guided me in the right direction. I also thank them for the questions they raised, each of which is critical now that I think about it.

I am extremely grateful to my parents for their love, caring and sacrifices in educating and preparing me for my future. Also, I express my thanks to my friends who continuously supported and encouraged me through the difficult time.

Contents

1 Introduction

Digital management systems are widely applied by libraries for storing, processing, querying, and disseminating information. The management systems consist of various books, scientific papers, music CDs and even videos. Union catalogues, which record collections of numerous libraries, are originally proposed to help the interlibrary loan between libraries. Librarians can query information through a single portal by using union catalogues. The Online Computer Library Center's (OCLC's) WorldCat is the most extensive cooperative catalogue ever, containing records of tens of thousands of libraries [1]. It provides platforms and tools for partner libraries to edit and retrieve records from their collections.

Retrieving an author's publications is an essential function of digital library management systems. The rapid increase of the publications triggers the interesting phenomenon that the authors of publications are different while they share the same name, especially when only name initials are provided, and these repeated author names reduced the precision of retrieval systems in digital libraries. Although retrieval efficiency has been improved by implementing fuzzy matching and logical search, the improvement of author retrieval, which aims to access publications belonging to particular authors, remains to be a key challenge. Besides, author-level indicators are crucial in bibliometric analysis as they can promote the understanding of the contributions of the individual author and can facilitate the analysis of research areas and publication trends. The traditional author name disambiguation works manually, requiring abundant time-consuming human work. Several institutions [2, 3] attempt to provide every author with a unique identifier, while the workload is beyond the institutional capability. Therefore, many publications are still lack of author identifiers. The authorship ambiguity becomes increasingly critical, and automatic authorship identification is required [4, 5].

1.1 Motivation

Previous research has demonstrated the application of both machine-learning-based (ML-based) and non-machine-learning-based (non-ML-based) approaches for author name disambiguation (AND) tasks [4, 5]. Non-ML-based approach mainly engage graph-based techniques [6, 7, 8, 9, 10, 11] and heuristic-based techniques [12, 13, 14, 15]. In addition, Caron and van Eck [16] proposed rule-based scoring to measure similarities between bibliographic records. With the advances in the development of computing abilities and research on modelling, machine learning techniques have been utilised to disambiguate authorships in complex bibliographic databases [17]. According to the usage of labelled training data, ML-based AND techniques can be divided into three categories: supervised techniques [18, 19, 20, 21, 22, 23, 24, 25, 26], unsupervised techniques [10, 27, 28, 29], and semi-supervised techniques [30, 31, 32]. Supervised AND techniques require reliable and representative labelled data to train the name disambiguation models [33]. Unsupervised AND techniques do not need extra labels and learn patterns by themselves [5]. Besides, semi-supervised AND techniques combine unlabelled data and a small amount of labelled data together to improve the performance of models [32].

Although several automatic approaches have been proposed in past years, they mainly focus on using a single schema for diverse data types to distinguish authorship and ignore the benefits of different features [34]. Usually, records (or publications) in bibliographic databases consist of textual features, discrete features and coauthor information. Textual features (such as titles, abstracts, etc.) of bibliographic records are often in the form of short and summative text and contain semantic information. The AND models have to understand and extract semantic information in the context. Discrete features are metadata of publications, including publisher, language, publication year, etc. The similarity of discrete features is a key indicator for authorship identification that multiple metadata is widely used to generate similarity profiles [17]. Coauthor information also has its own type of meaning, by which we can build coauthor networks and analyse the propinquity and 2-hop or 3-hop neighbours of coauthors to distinguish authors. Considering the three types of attributions inside bibliographic records, applying any specific technique of one type of data will waste information of other kinds of attributes. For example, [9] proposed a latent representation of the document, which combines two independent attributes, coauthorship information and word2vec embeddings of meta-content, and gained much better performance over various datasets. Therefore, we will consider data fusion and process the three types of data in their own ways.

Moreover, the majority of existing AND approaches are based on scientific papers rather than books [17]. Compared with books, scientific papers own consistent format and completing structure, including titles, abstracts, coauthor names, emails, venue names, affiliations, keywords, journal titles, etc. [35, 23]. However, book records in OCLC's WorldCat have less attributes, including title, coauthor names, publisher, country code, language, and publication year. Additionally, authors of book publications usually have fewer linkages with each other than authors of scientific papers, which is another difficulty of authorship identification among books.

Furthermore, uninterpretable neural networks are commonly used to achieve high performance for AND tasks over the past few years[24, 25]. However, applying uninterpretable models could cause both a practical and an ethical issue [36]. In recent years, the attention mechanism has been regarded as a tool to interpret the decision-making of neural networks. Although it is still controversial whether the attention mechanism provides reliable interpretations [37], to some extent, it can give intuitive explanations of neural architecture behaviours [38]. By giving each token of the input sequence different weights, the attention mechanism allows neural networks to concentrate on more critical parts when processing information with limited computing resources [39]. Therefore, attention mechanisms have become a crucial component of neural networks. For example, Bahdanau et al. [40] proposed an additive attention mechanism to emulate searching through a source sentence in decoders of machine translation models. Besides, the research by [41] proved the feasibility of applying the attention mechanism in sentiment classification tasks. In addition, Wang et al. [42] designed an effective attention-based transaction embedding model to recommend the next item according to existing transactions.

1.2 Research Goals and Questions

Given the limitations of current AND models that only one schema is used to process all data attributes, it would be interesting to incorporate all types of information in the bibliographic data, including textual features, discrete features and coauthor information. Meanwhile, to ensure information of all attributes is fully used, we have to choose the appropriate method for each attribute.

Due to the ability to explicitly force the model to focus on more critical parts of information, the attention mechanism is used to integrate embedded tokens of all three types of attributes. In addition, the attention mechanism is also used to visualise human-readable interpretations of decisions and help us conclude the importance of each attribute.

Based on the limitations in Section 1.1, we propose the following research questions for this thesis:

RQ1: Given various attributes as strings, how does a full-text model perform against a title-only model?

RQ2: Given various attributes as three kinds of features (textual, discrete, and coauthor features), how does an all-attribute model, which utilises all features with their corresponding techniques, outperform or underperform the text models?

RQ3: How do the attention mechanisms benefit the AND tasks?

(a) How does the attention-based model perform against the all-attribute models?

(b) How can the attention mechanisms provide evidence of model decisions?

(c) According to the attention weights, which attributes (e.g. title, coauthor names, languages, publication years, country codes, and publisher names) are more important than others in global scope?

The RQ1 aims to verify the performance of using all attributes as strings other than only using titles. The purpose of RQ2 is used to verify the improvements made by multiple kinds of features. The RQ3.a seeks to test the benefits of attention mechanisms. Its sub-questions RQ3.b and RQ3.c try to provide evidence of the model's decision by visualising attention weights and conclude the importance of each attribute in the bibliographic data by analysing the attention weights of each attribute.

1.3 Structure of Thesis

This thesis is organised in the following order. Chapter 2 introduces the necessary background knowledge used in this thesis, including the concepts of attention mechanisms, pre-trained language models, and graph embeddings. Chapter 3 elaborates the previous work from other authors. It contains five primary methods of AND tasks and related papers that fuse different types of information.

Chapter 4 introduces the original data and data processing workflows. Chapter 5 presents our proposed methodology for AND problems, including problem definitions, attention mechanisms, the proposed model and baseline models. Chapter 6 defines the experimental setup and evaluation metrics. Chapter 7 displays the results of the proposed and baseline models and analyses attention visualisations, followed by Chapter 8 which discusses the results with the research questions, limitations, and future work. Finally, Chapter 9 concludes the thesis and suggests directions for future work.

2 Background

This chapter introduces the background knowledge used in our proposed attention-based information fusion model for pairwise AND. The first section provides a brief introduction to previously proposed attention mechanisms. The second section gives an overview of pre-trained language models. The third section introduces the graph embeddings.

2.1 Attention Mechanism

Attention is a complex human cognition function. It is used to focus on elements, parts or details when detecting an object in a visual scene [43]. We can not only notice such elements by sensory stimulation (*BOTTOM-UP PROCESSING*), such as novelty and unexpectedness, but cognitive factors (*TOP-DOWN PROCESSING*), such as knowledge, expectation and current goal, can also draw our attention [44]. Inspired by the top-down control of attentional processes in the biological system of humans, attention has become a significant component of neural networks, focusing on distinctive parts when processing large amounts of information while providing interpretations of neural networks [39].

Attention was first proposed by Bahdanau et al. [40] to help memorise long input sentences in neural machine translation. Before that, researchers mainly built a sequence-to-sequence (seq2seq) model, which contains an encoder and a decoder, to solve neural machine translation tasks. The encoder's last hidden state produces a single context vector, which is also the decoder's input. The context vector is expected to contain all contextual information of the input sequence. Thus, a critical disadvantage of this context vector is the incapability of remembering long sentences. The attention mechanism proposed by Bahdanau et al. [40] creates shortcuts between the context vector and the entire input sequence, and the weights of these shortcuts are trainable for each output element. The new context vector in the attention mechanism involves three pieces of information: the encoder hidden states, the decoder hidden states, and the alignment between source and target elements. The context vector has full access to the entire source input so that no context meaning can be forgotten. There are already many different attention mechanisms proposed, such as additive [40], location-based

Table 2.1: Summary of score functions of attention mechanism.

Name	Alignment score function	Ref.
Additive	$score(s_t, h_i) = v_a^T tanh(W_a[s_t; h_i])$	[40]
Location-based	$score(s_t) = \text{softmax}(W_a s_t)$	[45]
General	$score(s_t, h_i) = s_t^T W_a h_i$	[45]
Concat	$score(s_t, h_i) = v_a^T \tanh(W_a[s_t; h_i])$	[45]
Dot-product	$score(s_t, h_i) = s_t^T h_i$	[45]
Scaled dot-product	$score(s_t, h_i) = \frac{s_t^T h_i}{\sqrt{n}}$	[46]
Similarity	$score(s_t, h_i) = \frac{s_t \cdot h_i}{\lvert\lvert s_t \rvert\rvert \cdot \lvert\lvert h_i \rvert\rvert}$	[47]

Note: s_t represents the t-th hidden state, h_i represents the i-th vector of the input sentence, n is the dimension of vectors, and W_a, v_a are all trainable matrices.

[45], general [45], dot-product [45], concat [45], scaled dot-product [46], similarity [47], global/soft [48], and local/hard [48, 45] attention mechanisms. The summary of score functions of the above attention mechanism is shown in Table 2.1.

Vaswani et al. [46] proposed a seq2seq model, which replaces the recurrent architecture with positional encoding, named Transformer, which is entirely built on the self-attention mechanisms. The multi-head self-attention mechanism inside Transformer is adopted from the scaled dot-product attention that the attention score is a weighted sum of the values, where the weight assigned to each value is determined by the dot-product of the query with all the keys. The scaled dot-product attention mechanism is formulated in the following:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{n}})\mathbf{V} \tag{2.1}$$

where **K** and **V** is a set of key-value pairs, both of dimension n and **Q** is a set of queries. The multi-head self-attention mechanism runs the above scaled dot-product attention multiple times in parallel. The output values are concatenated and projected, allowing the model to focus on information from different representation subspaces jointly.

Recently, attention mechanisms have been extended to process multimodal data. A typical method is the co-attention or cross-attention, which is first proposed to incorporate embeddings of two modalities in VilBERT [49] as:

$$\begin{cases} \mathbf{H}_A \leftarrow \mathrm{Attention}(\mathbf{Q}_B, \mathbf{K}_A, \mathbf{V}_A), \\ \mathbf{H}_B \leftarrow \mathrm{Attention}(\mathbf{Q}_A, \mathbf{K}_B, \mathbf{V}_B). \end{cases} \tag{2.2}$$

where $\mathbf{H}_A$ and $\mathbf{H}_B$ are outputs of two modality embeddings $\mathbf{X}_A$ and $\mathbf{X}_B$ after being processed by the co-attention module. The queries $\mathbf{Q}_A$, keys $\mathbf{K}_A$, and values $\mathbf{V}_A$ are generated by $\mathbf{X}_A$ and $\mathbf{Q}_B$, $\mathbf{K}_B$, and $\mathbf{V}_B$ are generated by $\mathbf{X}_B$. The co-attention mechanism can learn cross-modal interactions and does not increase the computational complexity [50].

Since 2018, the Pre-trained language model BERT, with the help of Transformer, has been widely used in many natural language processing (NLP) tasks [51]. However, it is still a black box with no reasonable way to interpret model decisions. Luo et al. [52] then proposed two attention mechanisms with BERT. The first one is the built-in self-attention of BERT. The second attention mechanism is a newly proposed BERT-Attention model, which adds an attention layer to the output of BERT to provide explanations for the BERT model by providing explicit weights to the classification layer. Experiments on question classification datasets show that the BERT-Attention model works better than the BERT model due to the explicit connections between the attention and classification layers. Meanwhile, the new mechanism improves the decision interpretation and helps identify the importance of tokens of the model's input.

2.2 Pre-trained Language Model

With the development of deep learning, neural networks are widely used in the field of natural language processing (NLP) [51]. Because deep neural networks usually have a large number of trainable parameters, the lack of labelled data has become a significant limitation of NLP methods. Therefore, pre-trained models (PTMs), which usually learn universal semantic information by themselves on large-scale unlabelled corpora, are proposed to improve the performance on downstream tasks by providing better word embeddings [53]. Because of the development of computing powers and the emergence of deep neural networks, the PTMs have become more advanced and own deeper architectures. According to the survey by Qiu et al. [54], the existing PTMs can be classified into two generations.

The first-generation PTMs focus on learning good word embeddings through shallow architectures with unlabelled data. Word2Vec is one of the most popular PTMs to generate task-free distributed representations of words. It can be utilised by two shallow architectures: Continuous Bag-of-Words (CBOW) and Skip-Gram (SG) models.

The CBOW model predicts the current masked word based on a window of surrounding context words, and the SG model predicts the surrounding window of context words given the current word [55, 56]. In addition, GloVe [57] is another widely used PTM, which trains on global word-word co-occurrence counts and simultaneously involves the meaningful linear substructures.

As the first-generation PTMs only provide context-free word-level embeddings, ignoring high-level concepts in context, the second-generation PTMs are proposed to generate contextual word embeddings and are tightly connected with downstream tasks. The first sentence or document encoder which produces contextual word embeddings is proposed by Dai and Le [58]. They presented two pre-training approaches to improve the following long short-term memory (LSTM) recurrent networks. The first approach predicts the next sequence, and the second approach is a sequence-to-sequence (seq2seq) model that encodes a sequence to a vector and decodes the vector to the original sequence again. The weights for the encoder network and the decoder network are the same. Peters et al. [59] proposed another popular encoder named ELMo, consisting of two layers of a bidirectional language model, for deep contextualised word representation. The ELMo can not only provide context-dependent semantic information and also model aspects of syntax. Therefore, the same word can be converted to different word embeddings in different sentences.

More recently, due to the success of natural language processing with Transformer, some very deep PTMs based on Transformer have shown their excellent performance in learning universal language representations [54]. OpenAI GPT (Generative Pre-training) uses a traversal-style approach and processes structured text input as a single contiguous sequence of tokens to realise task-specific input adaptations [60]. It avoids extensive architecture changes and enables efficient fine-tuning. In contrast, BERT (Bidirectional Encoder Representation from Transformer) proposed a bidirectional pre-training approach for language representations by using a ”masked language model” (MLM) pre-training objective, which enables jointly conditioning on both left and right contexts in all layers [61].

2.3 Graph Embeddings

In recent years, graph analysis has been widely used in computer science, and relevant areas, including social interactions and biological networks [62]. Graphs can efficiently store and access entities and their relations and are also able to provide features for machine learning tasks: (a) node classification, (b) link prediction, (c) clustering, and (d) visualisation [63]. The main purpose of graph analysis in machine learning tasks is to map high-dimensional structures into lower-dimensional embeddings. In the early

2000s, graph embeddings were used as dimensionality reduction techniques, and since 2010, scalable graph embedding techniques have been proposed to utilise network sparsity in real-world applications [63].

The traditional way to generate graph embeddings is factorisation-based methods, which aim to extract latent factors associated with vertices and use them in further machine learning models [64]. Typical factorisation based methods contains Locally Linear Embedding (LLE) [65], Laplacian Eigenmaps [66], and Graph Factorisation (GF) [64]. LLE maps its high-dimensional inputs into a single global coordinate system of lower dimensionality. LLE reconstructs global nonlinear structure by exploiting locally linear relations with neighbours in the embedding space. The Laplacian Eigenmaps algorithm represents the graph connections via the Laplacian matrix and aims to keep the embeddings of two neighbours close. GF is a large-scale and distributed graph decomposition and inference framework which can factorise graphs in parallel on hundreds of distributed computing resources.

With the development of graph embedding and NLP techniques, random walk based methods are commonly used to generate embedding vectors for each node according to its attributes and connections. They mainly convert complex network structures into random node sequences and use semantic methods in the NLP area to generate embeddings. DeepWalk [67], based on the language model Word2Vec [55], use local information from truncated random walks through the network by treating the node sequences as sentences. By taking nodes as the glossary, DeepWalk applies the Skip-Gram algorithm [55] to map nodes into the embedding space. By adding a biased-random walk to DeepWalk, *node2vec* [68] is proposed to provide a trade-off between breadth-first (BFS) and depth-first (DFS) graph searches. *node2vec* can achieve better performance than DeepWalk by accessing different neighbours.

2.4 Chapter Summary

In this chapter, we introduced the relevant background knowledge used in our thesis, including attention mechanisms, PTMs, and graph embedding techniques. We use PTMs to process textual features in authorship data and generate embeddings for each token. Similarly, the graph embedding techniques help process our coauthor information and generate node embeddings for each coauthor. To extract latent information from records, we apply self-attention to generate a latent feature vector for each record and the co-attention mechanism to extract interactive attention between two records. The detailed modules are further discussed in Chapter 4. In the next chapter, the existing related work of AND is discussed.

3 Related work

This chapter presents an introduction to related literature. The chapter firstly gives an overview of AND techniques using both ML and non-ML methods in general. Then it introduces related studies that use information fusion techniques to solve AND tasks.

3.1 Overview of Author Name Disambiguation

A bibliographic database contains a large number of publication records. Among these bibliographic records, there are two types of author name ambiguity. The first type is that the same author may have multiple names (synonymy or name variations), and the second one is that different authors may have the same name (polysemy or homonym) [4]. The AND tasks aim to distinguish authorship in bibliographic databases by existing record attributes. In other research fields, the term AND may have similar terms such as entity disambiguation, instance unification, authority control, web appearance disambiguation, semantic matching, record linkage, etc. [34].

According to the survey written by [34], the AND techniques can be classified based on the used methodology: (1) ML techniques, including supervised, unsupervised, and semi-supervised techniques specifically and (2) non-ML techniques, including graph-based techniques and heuristic-based techniques. The five types of AND techniques are described in the following subsections.

Supervised AND Techniques

The first strategy of calculating similarity scores of record pairs is supervised techniques which need ground-truth labels to train and evaluate models. Torvik and Smalheiser [18] proposed a model called "Author-ity", which calculates the probability that two scientific papers sharing the same author name were written by the same person using supervised learning. The dataset they used is MEDLINE [69] from the bibliographic database of the National Library of Medicine, and a 5-step procedure, including computing similarity profiles and estimating pairwise probability, was implemented for generating the trainable dataset. In addition, they improved the baseline model proposed in [70] by adding additional predictive features and supplemental information extracted from public web pages. Besides, clustering is used to generate author-individual clusters after similarity profiles are estimated.

Both Liu et al. [19] and Kim et al. [20] use supervised machine learning techniques to generate similarity profiles and apply additional pairwise classification to distinguish authors. They both test models using the PubMed database [69]. The major difference between the two papers is the way of calculating the similarity between two records. Liu et al. [19] applies weighted similarity score based on conventional inverse document frequency (IDF) [71] while Kim et al. [20] calculates cosine similarity of bag-of-words features.

More supervised AND techniques were proposed with the development of machine learning algorithms. The research by Han et al. [21] compares the performance among the extreme learning machine (ELM), support vector machines (SVM), and least squares support vector machines (LS-SVM) algorithms when applying to the AND tasks. They experiment with these three algorithms on two strategies, one classifier for each name (OCEN) and one classifier for all names (OCAN). The results show that both OCEN and OCAN strategies based on ELM have better generalisation performance and much faster learning speed than the two strategies based on SVM and LS-SVM algorithms. Wang et al. [22] proposed a boosted-trees AND method identifying authorship by pairwise classification. Rehs [23] conducted AND in 2017 Web of Science database using random forest and logistic regression algorithms. Descriptive statistics of metadata are used for similarity calculation for ML algorithms.

Recently, neural networks are also used in AND tasks. Deep neural networks, using string similarities, are tested in both English author names [24] and Vietnamese author names [25] datasets. Müller [26] introduced word embeddings to AND tasks. Word embeddings from Glove and word2vec are used to extract semantic information of title tokens, and a joint multi-layer neural network, inputting with semantic information and Jaccard similarity of tokens and coauthor names, produces the pairwise classification of records.

Unsupervised AND Techniques

Unlike supervised techniques, unsupervised techniques do not need manual ground-truth labels. It relies on predefined similarity measures or functions to find distinct author-individual clusters in ambiguous author names. Some unsupervised AND techniques have already been proposed in past years.

Tang and Walsh [10] developed a new algorithm using knowledge homogeneity scores, which is adopted from the concepts of cognitive maps in psychology and approximate structural equivalence in network analysis. If two authors sharing similar names have the same knowledge base, They should be recognised as the same author.

Here, approximate structure equivalent (ASE) is used to measure the similarity of knowledge bases. So that, authors within a cluster constructed by ASE should be more similar, which means the authors who share similar names in an ASE cluster are the same author. They achieved a higher level of accuracy but cost less time than common AND methods.

Tang et al. [27] proposed a unified probabilistic framework to distinguish ambiguous authorships, which is quite general that any relational features or local attributes can be processed in this framework. Specifically, they formalise the AND problem by a Hidden Markov Random Field (HMRF) model and take both content-based information and structure-based information as feature functions of the HMRF model. In the disambiguation process, the Bayesian information criterion is applied to determine the number of people K. Parameter estimation is done by a learning algorithm, which contains two iterative steps, Assignment of papers and Update of parameters. Due to interdependencies between paper assignments, which means that papers with similar content or strong relations tend to have the same label, their approach can improve the performance in AND tasks.

Another unsupervised AND technique is an unsupervised Dempster–Shafer theory (DST) based hierarchical agglomerative clustering algorithm proposed by Wu et al. [28]. The first step combines DST and Shannon's entropy to fuse disambiguation features like venue, coauthors, citations, and calculated co-relation similarities. Then they calculate the belief and plausibility of each author according to the fused information. Finally, a DST-based hierarchical agglomerative clustering algorithm is applied to a pairwise correlation matrix of papers. They claimed this approach is better than three unsupervised models and get comparable performance to a supervised model.

Qiao et al. [29] proposed a heterogeneous graph convolutional network embedding method to integrate multi-layer and heterogeneous relationships between publications and the node attribute information of publications and output a low-dimensional representation of each publication for AND. After generating publication representations, a graph-enhanced clustering method is implemented to accelerate the clustering speed. A significant advantage of the new clustering method is that the number of distinct persons and any other parameters are not required. Another advantage of the research is that the model can be continually retrained and process newly added publications incrementally.

Semi-supervised AND Techniques

Considering the need for a large amount of labelled data for supervised techniques and the challenge of determining the number of hidden clusters for unsupervised techniques, semi-supervised techniques are introduced to solve AND problems. It only utilises a small amount of labelled training data but incorporates a large amount of unlabeled data to improve performance.

Levin et al. [30] presented a self-supervised algorithm for AND in large bibliographic databases. First, they apply the standard bootstrapping approach from natural language processing tasks to construct initial clusters using high-precision features. And then, in the supervised stage, they implement standard supervised classifiers using these bootstrapped clusters. There are three classes of high-precision features. The first class includes common author and subject features like article titles, journal names, addresses, affiliations, subject categories, etc. The second class is features based on citations between articles. The third class is combinations of the above features to validate the importance of feature interactions. This approach works well in a large-scale dataset which contains 54,000,000 name instances.

Topic modelling is also a common chosen unsupervised method to self-generate raw clusters for later supervised machine learning. For example, Zhu and Li [31] proposed an enhancing object distinction model based on the probabilistic topic model and decision tree model. They firstly generate document topics by Latent Dirichlet Allocation (LDA), one of the probabilistic topic models, and then measure document similarities using the original attributes and generated topic features. The affiliation similarity and other context attribute similarities are combined to judge the authorship relations. Finally, a supervised BF-tree model is trained for the object distinction problem.

Additionally, topic modelling is also used in the multi-phase semi-supervised approach for AND is proposed by Zhao et al. [32] and utilises Microsoft academic search data to identify ambiguous names. This approach contains four stages. In the first stage, they set multiple rules to preprocess and separate the dataset and use the LDA topic modelling to construct an author-based network. In the second stage, community detection on the author-based network, as well as a self-taught model, is used to calculate raw disambiguation results. The third stage mainly focuses on improving the raw results in the second stage by generating training data for supervised SVMs. Finally, the raw and SVM results are ensembled in the fourth stage to generate the final results.

Graph-based AND Techniques

Graph network analysis of coauthor relationships is also critical for AND tasks. Graphs, where nodes represent author names and edges connecting them are coauthor relations, are natural representations of author name ambiguity problem [34]. After constructing co-authorship graphs, either network analysis methods or similarity measures are applied for disambiguation.

Fan et al. [6] proposed a graphical framework for name disambiguation named GHOST, which uses a co-authorship graph to calculate similarity scores and ignores all other attributes in metadata such as emails, titles, etc. The co-authorship graph is built with a novel, intuitive similarity formula. After that, message-passing techniques are used for producing author-individual clusters. Xu et al. [7] proposed a collective graph-based approach for author disambiguation. They create CoAuthor, CoTitle, CoOrganization, CoSummary, and CoVenue graphs by maximising the gap between positive and negative paper edges. A network embedding algorithm is introduced to generate an embedding vector for each paper based on jointly learning through the five graph networks. In the end, the clustering algorithm divides the papers sharing similar names into distinct author clusters for each ambiguous name. Km et al. [8] proposed two AND models called ATGEP and ATGEP-web. The ATGEP constructs the topic graph with edge pruning and also the author-similarity graph based on co-authorship information. After combining the two graphs, clustering will be implemented to get the results of authorship identification. The ATGEP-web is almost identical to ATGEP but adds external information and re-generates the final graph. Pooja et al. [9] utilise co-authorship graphs and output embeddings of word2vec on meta-content of documents to get the vector representation through a variational graph autoencoder for each document. After that, a neural network is used to calculate the final latent representation.

The above methods all implement clustering after graph construction. However, clustering is not mandatory, and the graph network itself is enough to distinguish ambiguous authors. For example, Shin et al. [11] introduced a graph model, GFAD, which is based on co-authorship information extracted from citations. They directly distinguish author name ambiguity by node splitting and merging with the namesake resolver and heteronymous name resolver, without clustering used.

Heuristic-based AND Techniques

Heuristic-based algorithms are proposed for quick solutions to avoid slow calculations. The solutions may not be optimal but at least near an optimal. Some researchers have already applied heuristic-based algorithms to AND tasks.

Varadharajalu et al. [12] introduced an AND technique which applies NetClus, an iterative algorithm for clustering heterogeneous star-schema information network, on the PubMed dataset. The first step breaks down affiliation strings into different components to directly match emails, URLs, and organisations to determine if multiple author names actually refer to one or more real-world authors. If they are not the same individual according to the direct matching, TF-IDF similarity of organisation components would be calculated, and Google's Geocoder extracts address components.

Besides, the coauthor network from Microsoft Academic Search is used to determine whether two same names refer to the same author or not. Finally, the individual clusters are generated by NetClus based on the above information.

Johnson et al. [13] developed an automatic extraction algorithm that can generate bibliographies of the target investigator using relevant institutional information. The proposed algorithm takes an investigator database containing descriptive information, for example, names and departmental affiliations, and the dataset of PubMed publications as input. Firstly, name-based queries are applied to extract all possible publications of the target investigator. Then, these candidate publications are clustered into separate identities. Finally, according to the word frequencies of the investigator's affiliation, journal name, title and keywords, the cluster with the highest similarity score is selected.

Pooja et al. [14] proposed an unsupervised heuristic-based approach along with incorporating extra web information. Based on the Jaccard similarity measurement, they created four different graphs: topic graph, co-authorship graph, reference author graph, and reference topic graph, respectively. Next, the proposed heuristic-based method generates a common graph from the four attribute graphs. Then, clustering is applied to the common graph, generating the intermediate clusters. Finally, with the help of the author's web page information and intermediate clusters, they got the final author-individual clusters.

A flexible, simple, generic, context-aware and effective multi-layer heuristics-based clustering framework is proposed by [15]. The first layer puts all papers with ambiguous author names into ambiguous blocks. The second layer incrementally clusters papers inside each block according to structure-aware features such as coauthors, affiliations and emails. The third layer continues to incrementally merge these clusters based on global features extracted by the Research2vec model, which is pre-trained as the word2vec algorithm on an abstract corpus. The global features are produced from textual features, including paper titles, abstracts, and keywords.

3.2 Processing and Integration of Mixed Types of Information in Author Name Disambiguation

For information of various types or multiple sources, processing and integration techniques are crucial for models. In previous research on author name disambiguation, most papers only process data and build models using one type of information. For example, YAMANI et al. [24] took all attributes as text so that distances between strings are calculated as features for later classification, and Xu et al. [7] constructed a network for each attribute named as CoVenue, CoSummary, CoOrganizatiom, CoTitle, and CoAuthor networks. These methods missed much hidden information involved in different attributes.

Recently, some integration techniques have been proposed for AND tasks. Levin et al. [30] integrated two independent kinds of attributes to generate aggregated features. Firstly, two independent features are extracted: (1) the author features from the article metadata and the subject features from the article metadata, and (2) citation features drawn from the Web of Knowledge database. Then the combinations of the above features are computed based on product conjunctions of every two features to produce higher-precision features. Then all features (author, citation, high-precision conjunction features) are fed to a supervised classifier.

Besides, the combination of sub-models, which process their aspects of information, is also used for AND tasks. Müller [26] proposed a binary classifier based on three auxiliary models, the semantic title model, surface title model, and simple coauthor model. Each of them takes a part of input attributes and represents an aspect of the classification problem. The semantic title model calculates the cosine similarity of word embeddings of title tokens from two authorship records. Both GloVe and word2vec are used to generate word embeddings and compute similarities. The surface title model mainly provides the string-matching feature for the two records. It calculates the content similarity of stemmed tokens, character 3-, 4-, and 5-grams, and word bigrams based on cosine and Jaccard similarity algorithms. The simple coauthor model calculates the cosine and Jaccard similarity of the normalised coauthor names. Finally, the outputs of the above models and all metadata attributes are fused by the joint model, a multi-layer neural network. Pooja et al. [9] proposed an unsupervised framework which leverages both the relational and non-relational aspects of the documents. They firstly create two independent graphs for two different dimensions, coauthor and meta-content. And then, two different variational graph autoencoders are applied respectively to the two graphs to embed the document representation for each record. Then the fusion representation, by concatenating both two representations from two autoencoders, is the input of a fully connected neural network layer for the final representation, which is used by hierarchical agglomerative clustering to identify authorship.

Additionally, Zhou et al. [72] combined the above two techniques (features aggregation and sub-models combination). They firstly generate the fusion features by simply concatenating five raw document features (i.e., coauthor, affiliation, venue, title, and keywords) and build a similarity graph based on these fusion features. Then another five similarity graphs are built based on the five raw document features. Next, they construct an encoder to integrate the six similarity graphs and a triplets decoder to feed the latent information to a multi-layer perception, which formulates the AND problem as a binary classification task.

Moreover, various types of information can be fused into a low-dimensional representation. For example, Qiao et al. [29] proposed a heterogeneous graph convolutional network embedding method incorporating multi-layer and heterogeneous relationships with random walks and negative sampling to generate a low-dimensional representation for each publication. The publication representations and the structure of the relationship network are used for a hierarchical agglomerative clustering to identify authorships.

3.3 Chapter Summary

Many AND approaches have been proposed in previous research, including ML techniques (supervised, unsupervised, and semi-supervised) and non-ML techniques (graph-based and heuristic-based). However, there are still the following limitations:

1. State-of-the-art pre-trained language models, such as BERT and GPT-3, have not been used to analyse textual features of bibliographic records.
2. Most papers only process one or two types of features and do not consider utilising all three data types (e.g. textual, tabular, and coauthor features) with their corresponding techniques.
3. Little research considers model interpretation, while decision logics behind the model are important for humans to rely on the disambiguation results.

4 Data

The training and evaluation of AND models need enough labelled datasets. In previous research, multiple open AND datasets [73] have been shared, such as Han-DBLP [74], Culotta-REXA [75], Wang-Arnetminer [76], Zhang-Aminer [35], Qian-DBLP [77], CustAND [78], LAGOS-AND [79], etc. These datasets contain features such as titles, abstracts, coauthor names, emails, venue names, affiliations, keywords, and journal titles. Compared with these datasets, the bibliographic records in library catalogues are less informative and contain fewer attributes related to authorship. For example, there is no abstract, emails, and affiliations in the Dutch Central Catalogue. Therefore, the above datasets do not satisfy our needs, and extracting labelled data from a real-world bibliographic database is necessary.

OCLC software manages metadata for library catalogues. These catalogues vary from 500 million bibliographic records and tens of thousands of institutes in the case of WorldCat to one small database with thousands of bibliographic records for one local library. We look at the Dutch Central Catalogue, a database with millions of bibliographic records for this assignment. The records are mainly from book publications, and some authors have ground-truth identifiers, meeting the requirements of our research questions.

The Dutch Central Catalog of OCLC follows a custom format PICA+. Fields are identified by the delimiter, a dollar sign [$] followed by four-digit codes (e.g., $021A represents the title statement of a record). Subfields are identified by the same delimiter followed by a letter (e.g., $a in the field of $021A represents the main title while $d represents the subtitle). An example of a record is shown in Appendix A and the description of all fields and subfields are available at OCLC's website[1].

We will first define the AND problem in mathematical formulas. Then, an overview of the Dutch Central Catalog database and the selected features are introduced. Last, two different datasets will be generated in two splitting methods.

[1] https://help.oclc.org/Metadata_Services/GGC/WorldCat_GGC/Conversie_pica3_MARC_21/Mappings_Pica_Pica3_MARC_21

4.1 Problem Definitions

We extract a dataset from the bibliographic database of OCLC, which consists of publication records. Each record contains at least the title of the book and a list of coauthors. Most records also contain metadata such as language, publication country, publication year, and name of the publisher. So, we propose these definitions:

- Author name: A string s_k containing only family names and first initials, such as "R. Smith".
- Author ID: a unique identity code a_i, representing a real-world author.
- Record: a record r_j in the bibliographic dataset, with a set of attributes.
- The dataset of all records: $\mathbf{R} = \{r_1, r_2, \dots, r_N\}$, where N is the total number of records.
- Name block: a record cluster $\mathbf{B}_k = \{\langle a_1^k, r_1^k \rangle, \dots, \langle a_i^k, r_j^k \rangle, \dots, \langle a_m^k, r_n^k \rangle\}$, where $1 \leq i \leq m, 1 \leq j \leq n$, including m distinct authors sharing the same author name, and n records. k represents the k-th author name.
- Record pair: a tuple of two records (r_i^k, r_j^k) in a name block $\mathbf{B}_k$.

4.2 Original Data

The DCC dataset provided by OCLC is dumped into a single text file containing more than 90 million records. Approximately 64 million of them have at least one Author ID. Specifically, there are about 9 million unique author names (initial-surname strings) and about 2.5 million unique Author IDs in total. Furthermore, only about 1.7 million unique author names have an Author ID. To sufficiently understand the original data, next, we shall explore the data statistically.

Firstly, we count the number of authors and the number of unique Author IDs for each book so that we can obtain the distribution of these values, as shown in Figure 4.1 and Figure 4.2. From the Figure 4.1, we can find that the number of authors per book varies. Most books have less than 60 authors, and the books with less than ten authors constitute a significant proportion (about 60%) of all books. However, not every author owns a unique identifier. As shown in Figure 4.2, most books have less 30 Author IDs.

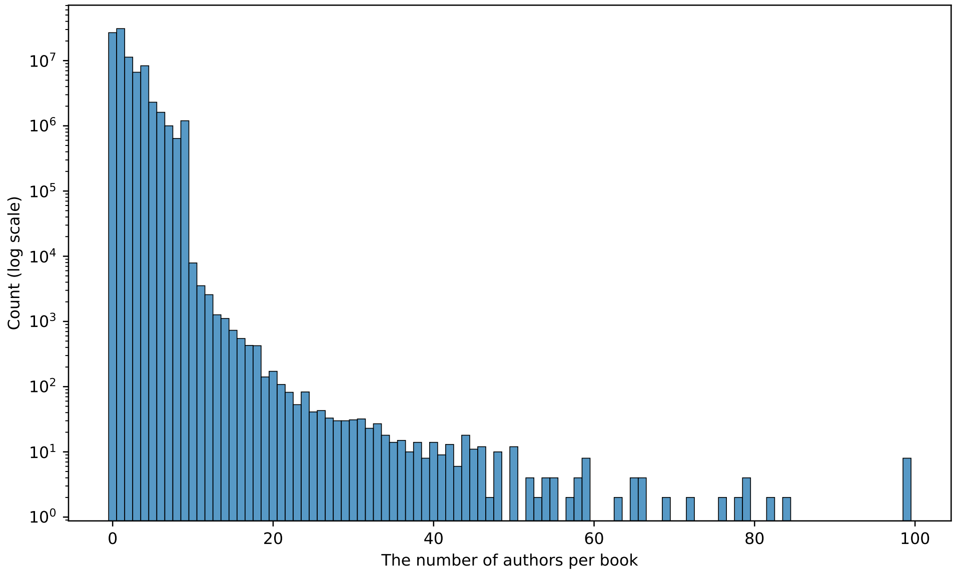


Figure 4.1: Distribution of the number of authors per book.

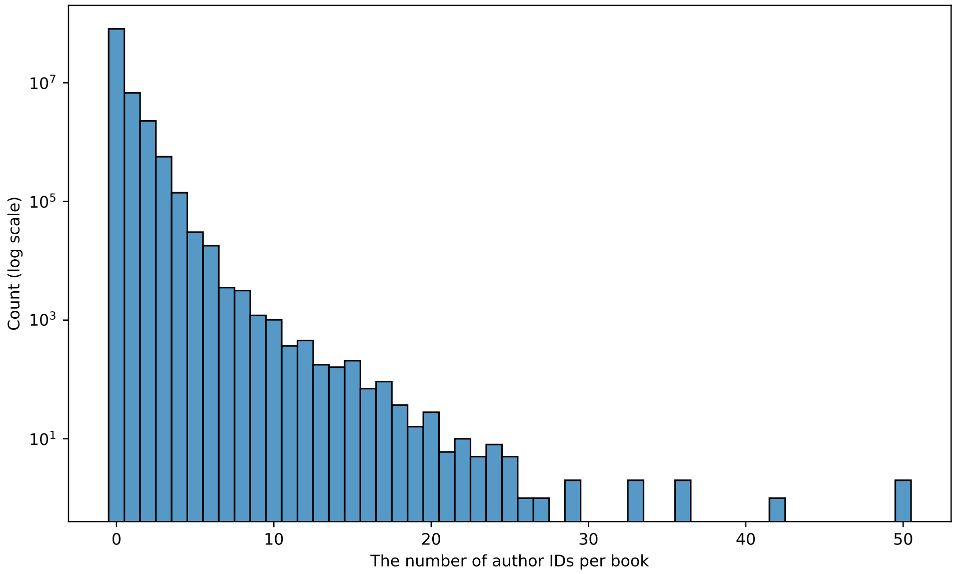


Figure 4.2: Distribution of the number of Author IDs per book.

Besides, the number of books written by each author is also calculated, as displayed in Figure 4.3 and Figure 4.4. The Figure 4.3 shows the distribution of the number of books written by each author name. For example, more than ten author names are connected with at least 10,000 books. The difference between the largest and smallest number of books belonging to each author name is so huge that the former number is 10 million times more than the latter. Furthermore, the distribution mainly follows Zipf's law [80]. Also, due to the limitation of Author ID annotations, the distribution of the number of books written by each Author ID shown in Figure 4.4 is different from the distribution in Figure 4.3.

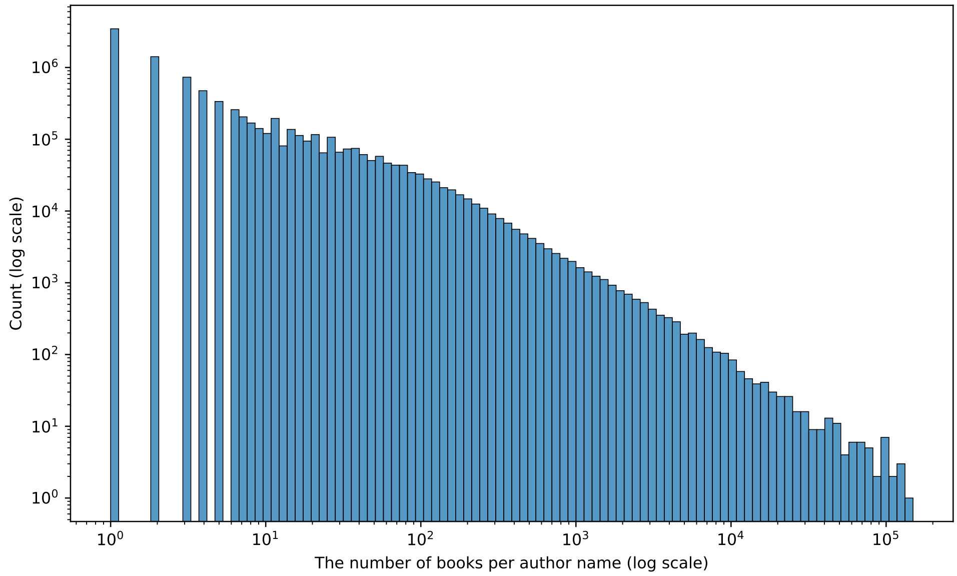


Figure 4.3: Distribution of the number of books per author name.

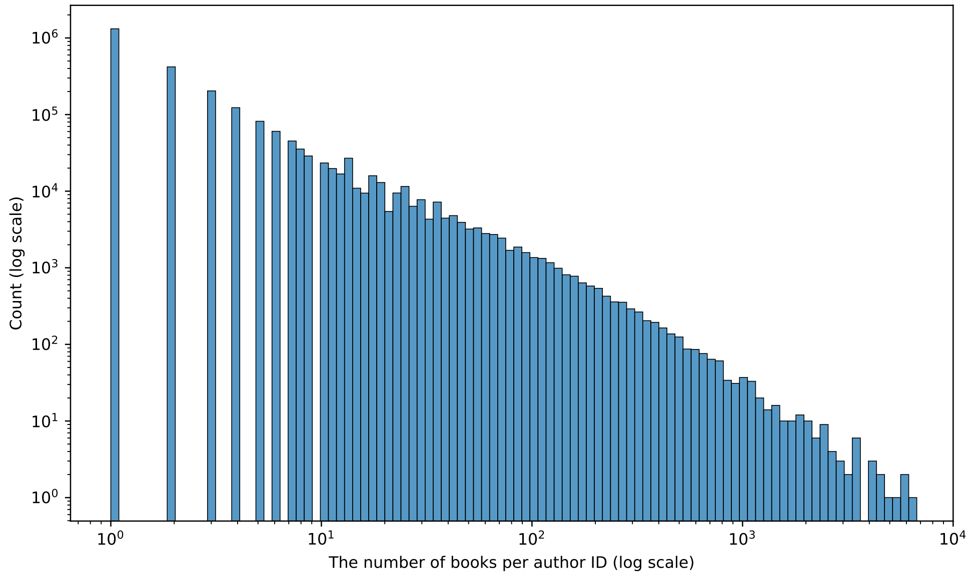


Figure 4.4: Distribution of the number of books per Author ID.

4.3 Feature Selection

Considering the relationship between all possible data fields and the name disambiguation task, we only select necessary fields according to OCLC database specialists. Besides, too sparse fields are ignored because they are useless in most cases. Therefore, as shown in Table 4.1, the final set of features contains title, coauthor, language, country code, publication year, and publisher name.

Table 4.1: Selected features of the final datasets.

Type	Feature Name	Description
Textual	Title	A string containing the title and subtitle of a publication
Metadata	Pub. Year	The year of publication
	Lang.	3-character code indicating the language of publication
	Country	2-character code indicating the place of publication
	Pub. Name	The full name of the publisher
Graph	Coauthor	A list consists of initial-surnames of coauthors

Most subfields are ignored to reduce the complexity of the feature space because a field usually contains multiple subfields. However, some subfields are remained by merging into the main field. For example, the main title and subtitle are concatenated to represent the full title.

4.4 Dataset Generation

Our disambiguation approach is based on record pairs in name blocks, where all papers share the same surname and initials of first names. Our proposed method tells that these record pairs are produced by the same or different authors. To sufficiently compare the proposed and baseline methods, we design two different dataset generation methods. The first method aims to simulate the real-world situation in that we match new records with existing labelled records in the database, and the models are trained using existing labelled records, so the train-test split is based on records. The second method is based on block splitting, which means that we split name blocks into training, test, and validation sets. The details are introduced in Section 4.4.1 and Section 4.4.2.

4.4.1 Splitting Method 1: Record-Splitting

The record splitting method firstly preprocesses data and then selects the top N name blocks as datasets, as shown in Figure 4.5.

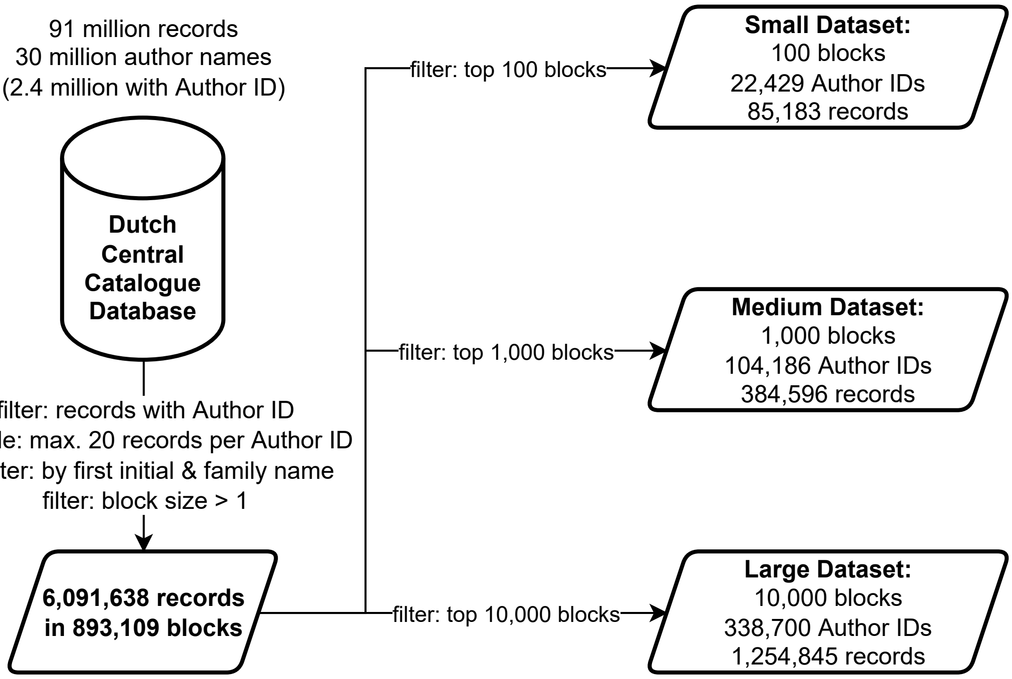


Figure 4.5: Data processing workflow of the record-splitting method.

Firstly, records without an author identifier (Author ID) should be removed because identifiers are necessary for training and evaluation. Next, we only randomly retain at most 20 books for each Author ID to cover more publications with limited computing resources and allow models to gain better generalization ability rather than only focusing on top authors. In addition, we need to merge name variations of authors into name blocks. For example, "Robert Smith" and "R. Smith" should be put into the same name block. Thus, we only consider the family name and initials of the first name to ignore name variations. After that, we will remove author names connected with only one record, which means we cannot generate record pairs in this name block. To test the proposed model's generalizability ability, we then put top name blocks into small, medium, and large datasets with the first 100, 1000, and 10000 blocks, respectively. The top name blocks represent the author names which produce the most publications.

In the real world, after getting a new record, we usually predict the relationships between the new record with existing records that share the same author name in the database, so dataset splitting should be based on records. As displayed in Figure 4.6, the dataset splitting has three steps:

1. For every dataset (small, medium, and large), 10% of records are selected as

validation records, and another 10% are test records. Thus the left 80% of records are training records.

2. We construct the training set by pairing training records. The validation and test sets are constructed specially. One record in each validation sample is obtained from the training records to represent existing records. The other record is chosen from validation records to represent new records. The generation of the test set is similar to the validation set.

3. After the training, validation, and test sets are generated, negative pairs, which means the two records are not written by the same author, are randomly sampled to the same number as positive samples to balance the final sets.

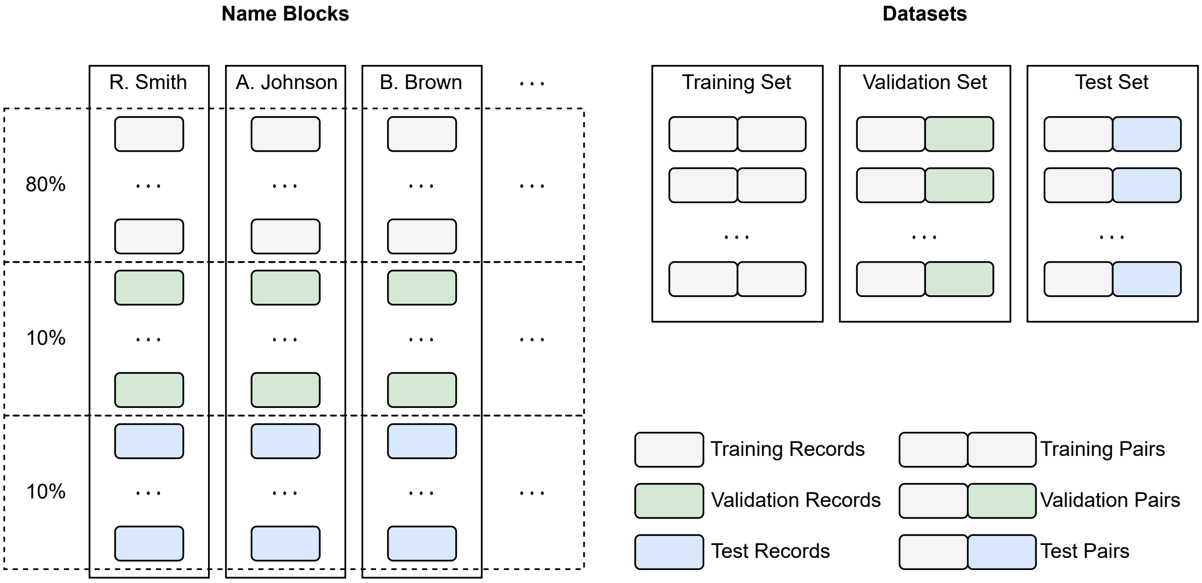


Figure 4.6: Dataset splitting workflow of the record-splitting method.

The statistical information of the three final datasets is shown in Table 4.2. The # *Blocks* represents the number of name blocks (initial-surnames) in each dataset, and the # *Books* means the number of books. Similarly, # *Authors* represents the number of distinct Author IDs in each dataset. The training, validation, and test subsets are constructed by combining records in each dataset. The # *Pairs* and *Pct.* columns introduce the number of record pairs in each subset and its percentage. In addition, the splitting is based on records, so we can only control the percentage of training, validation, and test records, but not the percentages of training, validation, and test pairs. Last, because we randomly select negative record pairs according to the number of positive record pairs, the values of # *Pos.* and # *Neg.* columns in each row should be the same.

Table 4.2: Statistical information of record splitting datasets.

Set	# Blocks	# Records	# Authors	Subset	Pct.	# Pairs	# Pos.	# Neg.
S.	100	85,183	22,429	train	66.63%	497,872	248,936	248,936
				valid	16.73%	125,048	62,524	62,524
				test	16.64%	124,316	62,158	62,158
M.	1,000	384,596	104,186	train	66.71%	2,286,348	1,143,174	1,143,174
				valid	16.59%	568,688	284,344	284,344
				test	16.70%	572,276	286,138	286,138
L.	10,000	1,254,845	338,700	train	66.69%	8,075,562	4,037,781	4,037,781
				valid	16.62%	2,012,374	1,006,187	1,006,187
				test	16.69%	2,020,766	1,010,383	1,010,383

4.4.2 Splitting Method 2: Block-Splitting

As shown in Figure 4.7, the data preprocessing workflow of the block-splitting method is almost the same as that of the record-splitting method except for shuffling all the name blocks to split sets randomly. The test and validation sets are fixed. The test set is chosen from the first 1000 name blocks, and the validation set consists of the 1001-2000 name blocks. Besides, we construct four different-sized training sets to investigate models' performance under different training sets.

Table 4.3: Statistical information of block splitting datasets.

Dataset	# Blocks	# Records	# Authors	# Pairs	# Pos.	# Neg.
Test	1,000	53,162	12,829	515,586	257,793	257,793
Valid	1,000	51,819	12,479	520,716	260,358	260,358
Small.Train	100	4,786	1,187	45,994	22,997	22,997
Medium.Train	1,000	50,143	12,228	486,732	243,366	243,366
Large.Train	10,000	437,731	107,273	4,377,174	2,188,587	2,188,587
Extra-Large.Train	100,000	2,074,432	505,122	22,997,808	11,498,904	11,498,904

The record pair generation of the block-splitting method differs from that of the record-splitting method. We generate all possible record pairs by combining every two records inside each name block and undersample negative pairs to the number of positive pairs. Similar to the record splitting datasets, the statistical information of block splitting datasets is shown in Table 4.3.

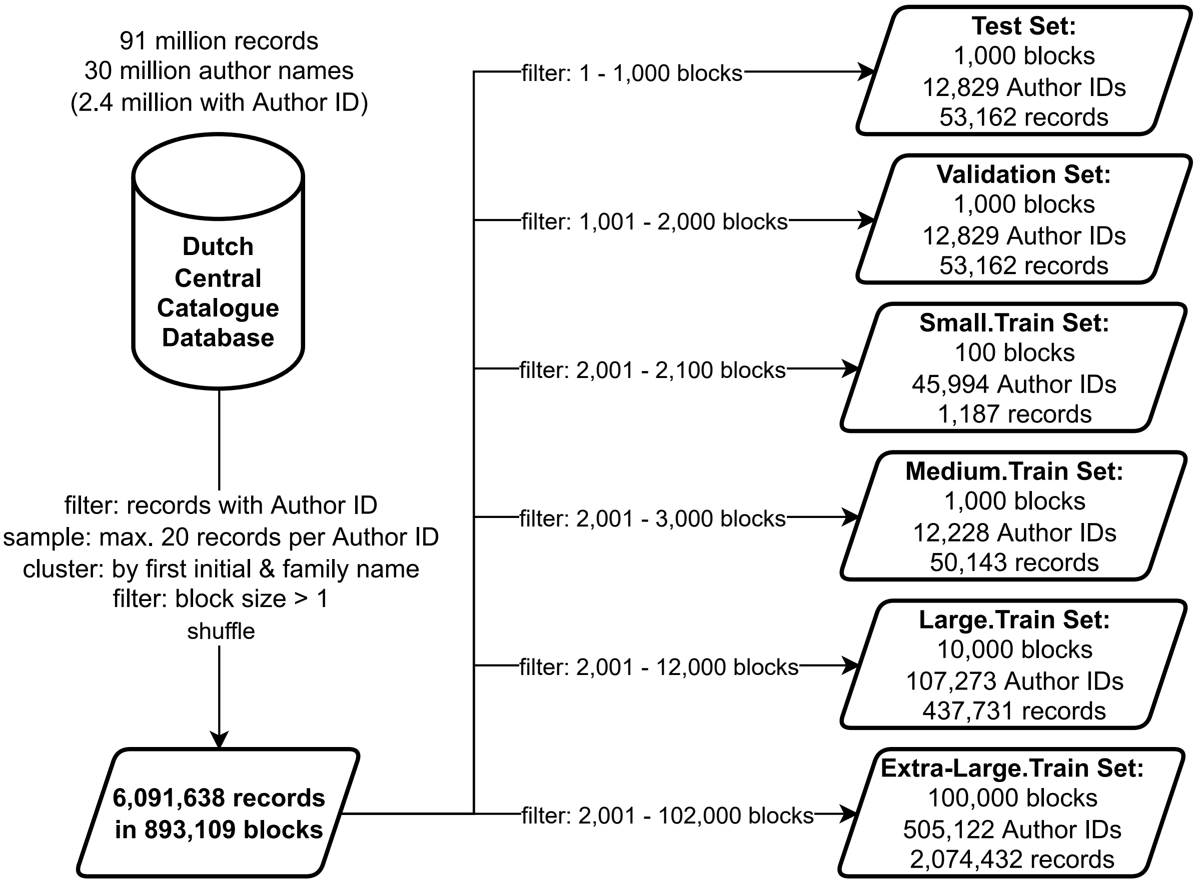

91 million records
30 million author names
(2.4 million with Author ID)
Dutch Central Catalogue Database
filter: records with Author ID
sample: max. 20 records per Author ID
cluster: by first initial & family name
filter: block size > 1
shuffle
6,091,638 records in 893,109 blocks
filter: 1 - 1,000 blocks
Test Set: 1,000 blocks 12,829 Author IDs 53,162 records
filter: 1,001 - 2,000 blocks
Validation Set: 1,000 blocks 12,829 Author IDs 53,162 records
filter: 2,001 - 2,100 blocks
Small.Train Set: 100 blocks 45,994 Author IDs 1,187 records
filter: 2,001 - 3,000 blocks
Medium.Train Set: 1,000 blocks 12,228 Author IDs 50,143 records
filter: 2,001 - 12,000 blocks
Large.Train Set: 10,000 blocks 107,273 Author IDs 437,731 records
filter: 2,001 - 102,000 blocks
Extra-Large.Train Set: 100,000 blocks 505,122 Author IDs 2,074,432 records

Figure 4.7: Dataset process workflow of the block-splitting method.

5 Methodology

This chapter gives details of our proposed methodology. In the first section, we propose the co-attention-based AND model and the co-attention and self-attention mechanisms for pairwise learning. Next, we describe two baseline models incorporating three kinds of features using their corresponding methods. The last section defines two baseline text models.

5.1 Co-Attention-Based AND Model

This section proposes the novel co-attention-based AND model. As shown in Figure 5.1, the model accepts record pairs as input and predicts the probability that they are written by the same author or not. Each record pair consists of two records in the same name block, and each bibliographic record contains three kinds of attributes from the publication metadata.

The first attribute is textual features, such as titles, which provide the main topic and cover a minimal summary of the contents. The best method to process textual features is PTMs, which use various statistical and probabilistic approaches to understand text data according to its context and output embeddings for each token and the whole sentence. By applying PTMs to the textual features, we can calculate context embeddings for every token of record titles.

To accelerate the training process and utilise more training data, we use an optional random projection [81] to project high-dimensional BERT embeddings (768) into a lower-dimensional subspace (128). Therefore, the intermediate BERT output can be cached for future use. This is done by a random projection matrix $\mathbf{R} \in \mathbb{R}^{k \times d}$with $\mathbf{R}(i,j) = r_{ij}$, where k is the original dimension, d is the target dimension, and $\{r_{ij}\}$ are independent random variables:

$$r_{ij} = \begin{cases} +1 & \textit{with probability } 1/2, \\ -1 & \textit{with probability } 1/2. \end{cases} \tag{5.1}$$

The second essential component of bibliographic records is structural data, including languages, publication years, country codes, and publisher names. Usually, encoding

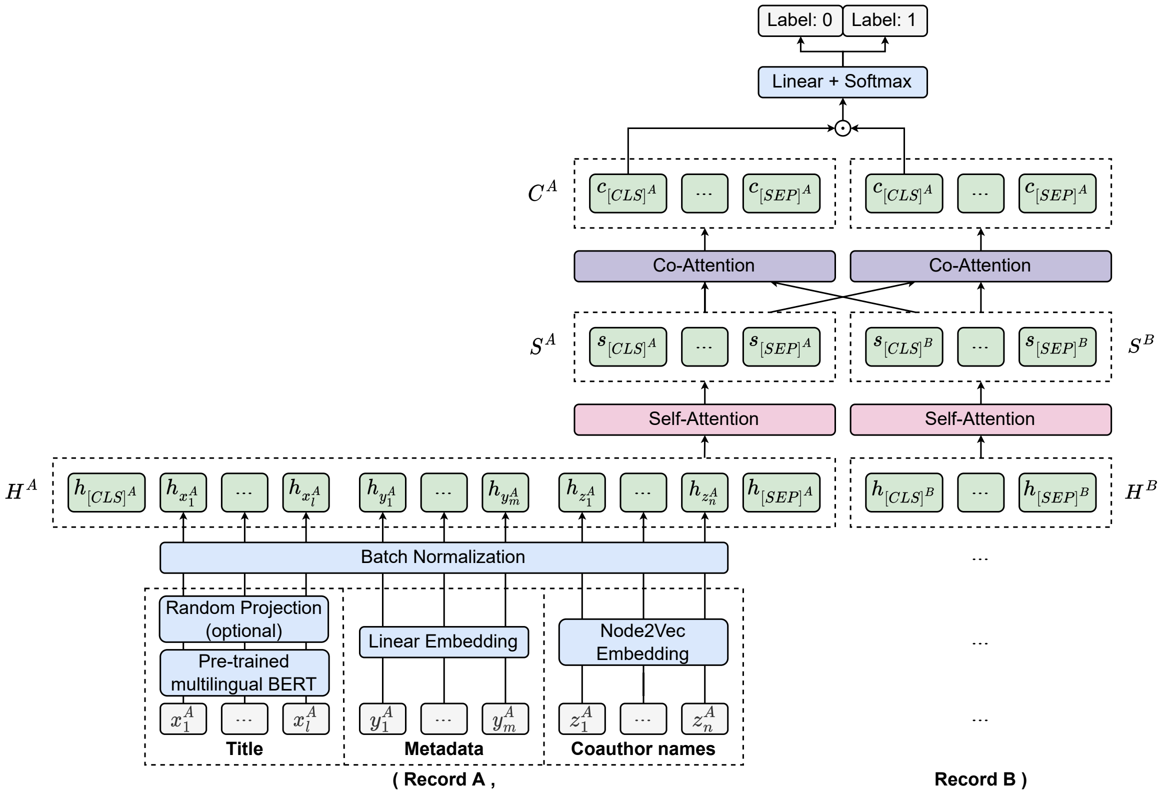


Figure 5.1: The structure of the co-attention-based AND model.

techniques are implemented to tabular data for machine learning models. Because we need to concatenate embeddings of different types of data, a linear embedding layer is added to map coded values to the same dimension of context vectors of titles.

The last type of data is coauthor information stored in an undirected network, where each node represents a distinct author name. We can extract author embeddings for each coauthor of a record by relative graph embedding techniques. We apply *node2vec* here for its flexibility and scalability. Moreover, it can be pre-trained on the coauthor network and provide embeddings for each initial-surname string of coauthors.

After getting embeddings from the above three types of data, we can concatenate (1) output embeddings of BERT for each token, (2) embeddings of each metadata column (structural data), and (3) embeddings of each coauthor through the *node2vec* module, into a hidden context matrix. Because the three parts of embeddings are generated by three different modules, their vector spaces can differ considerably. Thus, we apply a batch normalisation [82] layer to perform normalisation on all vectors for each training mini-batch. The calculation of normalisation is formulated as:

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \mathrm{E}[\mathbf{x}]}{\sqrt{\mathrm{Var}[\mathbf{x}] + \epsilon}}, \tag{5.2}$$

where $\mathbf{x}$ is a vector in the hidden context matrix, and ϵ is a value for numerical stability.

Let d be the dimension of hidden vectors after the batch normalisation layer, l be the max length of title tokens of each record, m be the number of metadata columns of each record, n be the max number of coauthors of each record. $\mathbf{X}^A \in \mathbb{R}^{l \times d}$ is a matrix consisting hidden vectors $\{\mathbf{h}_{\mathbf{x}_1^A}, \mathbf{h}_{\mathbf{x}_2^A}, \ldots, \mathbf{h}_{\mathbf{x}_l^A}\}$ that the BERT produced for the input tokens of the first record's title, and the matrix $\mathbf{X}^B \in \mathbb{R}^{l \times d}$, which consists of $\{\mathbf{h}_{\mathbf{x}_1^B}, \mathbf{h}_{\mathbf{x}_2^B}, \ldots, \mathbf{h}_{\mathbf{x}_l^B}\}$, is produced by the BERT for the tokens of the second record's title. Similarly, the hidden matrix of columns can be represented as $\mathbf{Y}^A \in \mathbb{R}^{m \times d}$ and $\mathbf{Y}^B \in \mathbb{R}^{m \times d}$, and that of coauthors can be formulated as $\mathbf{Z}^A \in \mathbb{R}^{n \times d}$ and $\mathbf{Z}^B \in \mathbb{R}^{n \times d}$. As a matter of convenience, we represents all hidden context vectors of each record as $\mathbf{H}^A \in \mathbb{R}^{N \times d}$ consisting of $\{\mathbf{h}_1^A, \mathbf{h}_2^A, \ldots, \mathbf{h}_N^A\}$ and $\mathbf{H}^B \in \mathbb{R}^{N \times d}$ consisting of $\{\mathbf{h}_1^B, \mathbf{h}_2^B, \ldots, \mathbf{h}_N^B\}$, where $N = l + m + n + 2$. $\mathbf{H}^A$ and $\mathbf{H}^B$ are constructed by concatenating the above embeddings, as formulated as:

$$\mathbf{H}^A = \begin{bmatrix} \mathbf{h}_{[\mathrm{CLS}]^A} \\ \mathbf{X}^A \\ \mathbf{Y}^A \\ \mathbf{Z}^A \\ \mathbf{h}_{[\mathrm{SEP}]^A} \end{bmatrix}, \mathbf{H}^B = \begin{bmatrix} \mathbf{h}_{[\mathrm{CLS}]^B} \\ \mathbf{X}^B \\ \mathbf{Y}^B \\ \mathbf{Z}^B \\ \mathbf{h}_{[\mathrm{SEP}]^B} \end{bmatrix}, \tag{5.3}$$

where $\mathbf{h}_{[\mathrm{CLS}]^A}$, $\mathbf{h}_{[\mathrm{SEP}]^A}$, $\mathbf{h}_{[\mathrm{CLS}]^B}$, and $\mathbf{h}_{[\mathrm{SEP}]^B}$ are special vectors, with dimension d, which contain no information itself. They are produced by a distinct linear embedding layer which convert tokens $\{[\mathrm{CLS}]^{\mathrm{A}}, [\mathrm{SEP}]^{\mathrm{A}}, [\mathrm{CLS}]^{\mathrm{B}}, [\mathrm{SEP}]^{\mathrm{B}}\}$ into $\{\mathbf{h}_{[\mathrm{CLS}]^A}, \mathbf{h}_{[\mathrm{SEP}]^A}, \mathbf{h}_{[\mathrm{CLS}]^B}, \mathbf{h}_{[\mathrm{SEP}]^B}\}$. A [CLS] token is used to involve all information in the sentence and can be utilised for prediction later. And the [SEP] tokens are marks for the ending of sequences.

In order to detect the importance of each vector in the hidden matrices $\mathbf{H}^A$ and $\mathbf{H}^B$, we first propose to design a self-attention module that captures interactive connections among all features inside a record, as formulated as:

$$\begin{aligned} \mathbf{S}^A &= \mathrm{SelfAttn}(\mathbf{H}^A), \\ \mathbf{S}^B &= \mathrm{SelfAttn}(\mathbf{H}^B), \end{aligned} \tag{5.4}$$

where $\mathbf{S}^A \in \mathbb{R}^{N \times d}$ and $\mathbf{S}^B \in \mathbb{R}^{N \times d}$ are representation matrices of each records. Interactive connections between tokens are calculated in the self-attention modules.

After that, the co-attention modules are proposed to calculate interactive attention between the two records:

$$\begin{aligned} \mathbf{C}^A &= \mathrm{CoAttn}(\mathbf{H}^A, \mathbf{H}^B), \\ \mathbf{C}^B &= \mathrm{CoAttn}(\mathbf{H}^B, \mathbf{H}^A) \end{aligned} \tag{5.5}$$

The details of the self-attention and co-attention modules are introduces in Section 5.1.1 and Section 5.1.2, respectively.

We took the output $\mathbf{c}_{[CLS]^A} \in \mathbb{R}^d$ and $\mathbf{c}_{[CLS]^B} \in \mathbb{R}^d$ from $\mathbf{C}^A$ and $\mathbf{C}^B$ as the holistic representations of the two records. The overall representation $\mathbf{r} \in \mathbb{R}^d$ of the record pair is computed by an element-wise product between $\mathbf{h}_{[CLS]^A}$ and $\mathbf{h}_{[CLS]^B}$. It is provided to a fully connected layer followed by a softmax layer to calculate the probability of the record pair written by the same or different authors. The resulting probability can be represented as a vector $\mathbf{o} \in \mathbb{R}^{d_o}$, where d_o equals the number of classes, which should be 2 for pairwise AND tasks. The first value in $\mathbf{o}$ represents the probability of the two records written by the same author. The order value is the probability of the two records written by different authors. The two probabilities should be added to 1. The linear and softmax layer can be formulated as:

$$\mathbf{r} = \mathbf{c}_{[CLS]^A} \odot \mathbf{c}_{[CLS]^B}, \tag{5.6}$$

$$\mathbf{o} = \text{softmax}({\mathbf{W}_o}^T \mathbf{r} + \mathbf{b}_o), \tag{5.7}$$

where $\mathbf{W}_o \in \mathbb{R}^{d \times d_o}$and $\mathbf{b}_o \in \mathbb{R}^{d_o}$ are parameters to be learned during training.

5.1.1 Self-Attention Modules

The self-attention modules are two independent scaled dot product attention blocks that can directly link two arbitrary elements in a sentence so that distant elements can interact through shorter paths. In our case, the two attention blocks allow elements of different kinds of attributes to interact with each other. They can overcome the limitations imposed by different feature spaces of different attributes. Figure 5.2 shows the details of a self-attention module.

With one of the hidden context matrices $\mathbf{H} \in \mathbb{R}^{N \times d}$, consisting of N vectors $\{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_N\}$ from the embeddings from titles, metadata, and coauthors, we can obtain queris $\mathbf{Q} \in \mathbb{R}^{N \times d}$, keys $\mathbf{K} \in \mathbb{R}^{N \times d}$ and values $\mathbf{V} \in \mathbb{R}^{N \times d}$ by linear projections of $\mathbf{H}$, which can be formulated as:

$$\begin{aligned} \mathbf{Q} &= \mathbf{W}_Q \mathbf{H}, \\ \mathbf{K} &= \mathbf{W}_K \mathbf{H}, \\ \mathbf{V} &= \mathbf{W}_V \mathbf{H}, \end{aligned} \tag{5.8}$$

where $\mathbf{W}_Q \in \mathbb{R}^{d \times d}$, $\mathbf{W}_K \in \mathbb{R}^{d \times d}$, and $\mathbf{W}_V \in \mathbb{R}^{d \times d}$ are trainable parameter matrices. Then the self-attention modules taks $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ as input and computes attention weights $\mathbf{self\_attn} \in \mathbb{R}^{N \times N}$ for the input sentence:

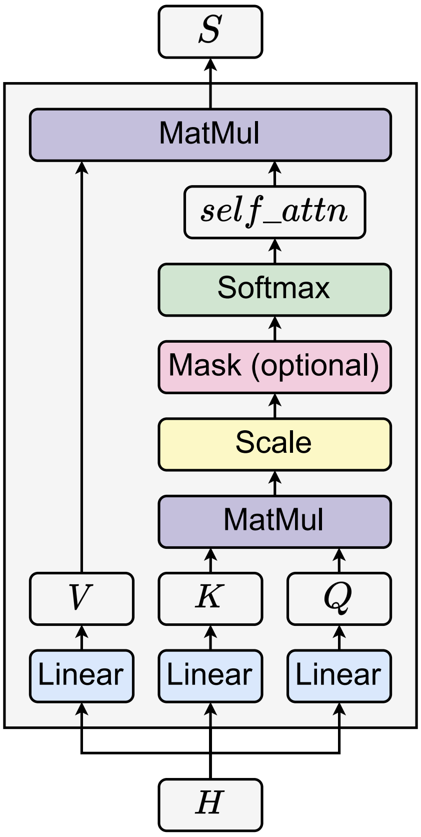


Figure 5.2: The self-attention module.

$$\textbf{self\_attn} = \text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d}}) \tag{5.9}$$

Thus, the representation matrix of a record **S** can be easily calculated by multiplying the self-attention weights **self_attn** and the values **V**, as formulated as:

$$\begin{aligned} \mathbf{S} &= \textbf{self\_attn} \cdot \mathbf{V} \\ &= \text{softmax}(\frac{\mathbf{QK}^T}{\sqrt{d}})\mathbf{V} \end{aligned} \tag{5.10}$$

5.1.2 Co-Attention Modules

Co-attention methods are proposed to solve multimodal problems, aiming to extract interactive connections between modalities [50]. We adopt the co-attention mechanism to AND tasks to explicitly provide evidence for the connections of two items, as shown in Figure 5.3. The adopted co-attention mechanism is based on the scaled dot-product attention, containing queries, keys and values. However, in order to find out interactive connections between two records, we need to generate queries $\mathbf{Q}^A, \mathbf{Q}^B \in \mathbb{R}^{N\times d}$, keys $\mathbf{K}^A, \mathbf{K}^B \in \mathbb{R}^{N\times d}$, and values $\mathbf{V}^A, \mathbf{V}^B \in \mathbb{R}^{N\times d}$ for the two records by linear transforms

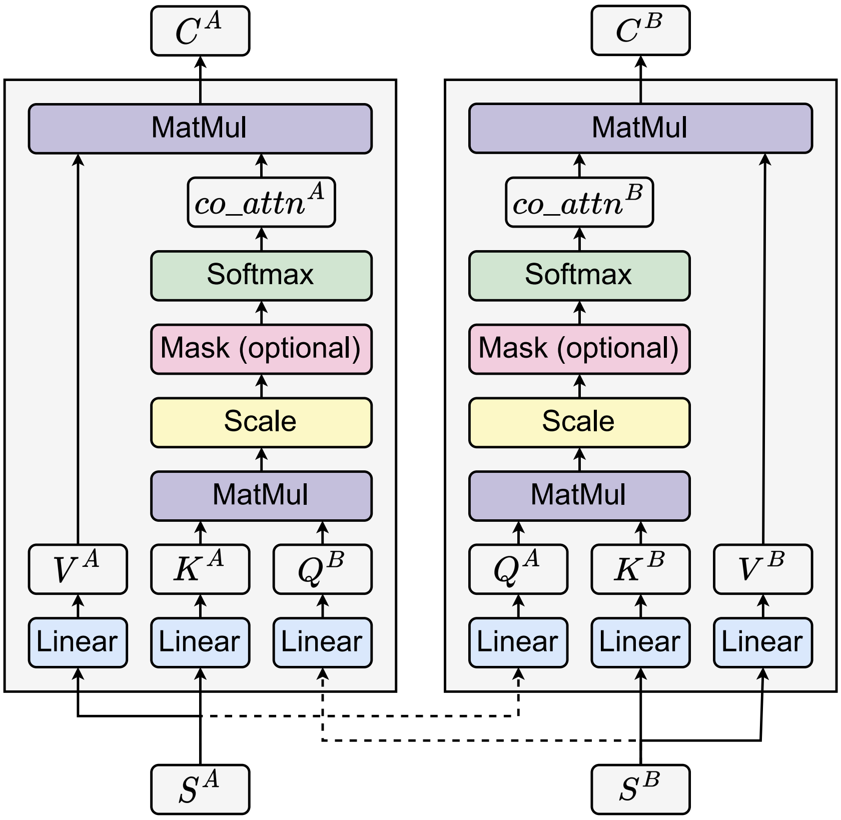


Figure 5.3: The co-attention modules.

of $\mathbf{H}^A$ and $\mathbf{H}^B$, which can be formulated as:

$$\begin{aligned} \mathbf{Q}^A &= \mathbf{W}_Q^A \mathbf{H}^A,\ \mathbf{Q}^B = \mathbf{W}_Q^B \mathbf{H}^B, \\ \mathbf{K}^A &= \mathbf{W}_K^A \mathbf{H}^A,\ \mathbf{K}^B = \mathbf{W}_K^B \mathbf{H}^B, \\ \mathbf{V}^A &= \mathbf{W}_V^A \mathbf{H}^A,\ \mathbf{V}^B = \mathbf{W}_V^B \mathbf{H}^B. \end{aligned} \tag{5.11}$$

The co-attention mechanism firstly calculates attention weights $\mathbf{co\_attn}^A, \mathbf{co\_attn}^B \in \mathbb{R}^{N \times N}$ for each input sentence as:

$$\begin{aligned} \mathbf{co\_attn}^A &= \mathrm{softmax}(\frac{\mathbf{Q}^B \mathbf{K}^{A^T}}{\sqrt{d}}), \\ \mathbf{co\_attn}^B &= \mathrm{softmax}(\frac{\mathbf{Q}^A \mathbf{K}^{B^T}}{\sqrt{d}}) \end{aligned} \tag{5.12}$$

Then, the final representations $\mathbf{C}^A$ and $\mathbf{C}^B$ are generated by multiplying the values $\mathbf{V}^A, \mathbf{V}^B$ and their attention weights as:

$$\begin{aligned} \mathbf{C}^A &= \mathbf{co\_attn}^A \mathbf{V}^A, \\ \mathbf{C}^B &= \mathbf{co\_attn}^B \mathbf{V}^B \end{aligned} \tag{5.13}$$

5.2 Baseline Full-attribute Models

The first baseline model is the mean-based AND model that incorporates all context vectors of the two records with a batch normalization layer, as shown in Figure 5.4. Let the hidden context matrix be $\mathbf{H} \in \mathbb{R}^{2N \times d}$, consisting of $\{h_1, h_2, \dots, h_N\}$ and covering all context vectors from titles, metadata, and coauthor information. Here, $N = l + m + n$ because no special token is used. The joint representation $\mathbf{r} \in \mathbb{R}^d$ of the two records is calculated as:

$$\mathbf{r} = \text{mean}(\mathbf{H}) \tag{5.14}$$

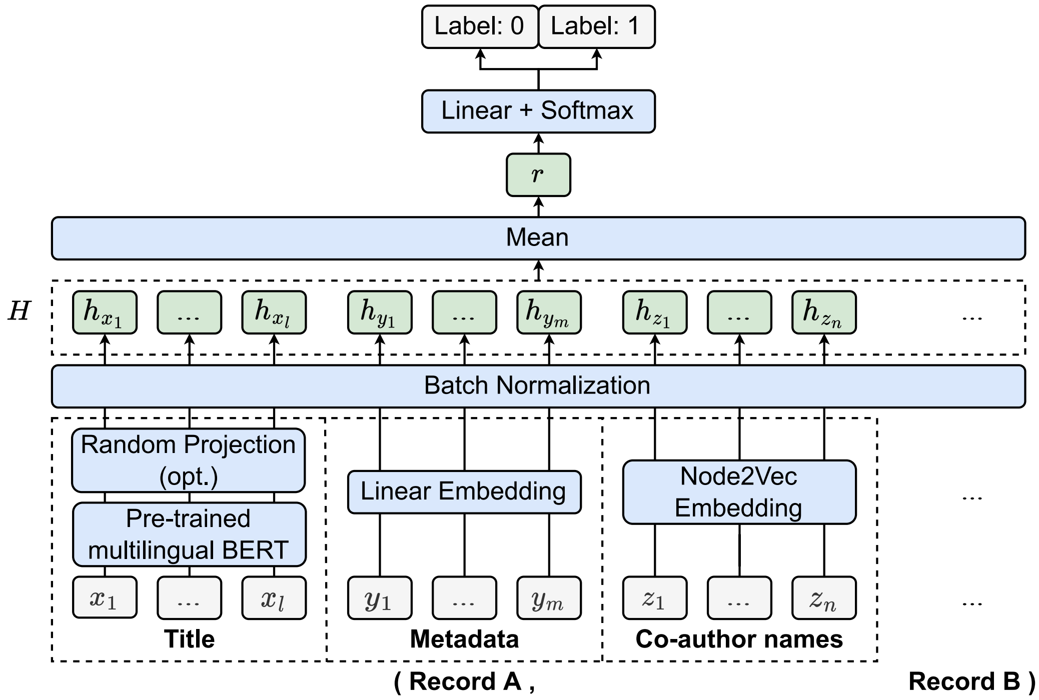


Figure 5.4: Baseline mean-based AND model.

The final predicted probability is generated by a fully connected layer and a softmax layer. The calculation of the output $\mathbf{o} \in \mathbb{R}^{d_o}$ is formulated as:

$$\mathbf{o} = \text{softmax}(\mathbf{W}_o{}^T \mathbf{r} + \mathbf{b}_o), \tag{5.15}$$

where $\mathbf{W}_o \in \mathbb{R}^{d \times d_o}$ and $\mathbf{b}_o \in \mathbb{R}^{d_o}$ are parameters to be learned during training.

The second baseline model is the linear-based AND model, adopted from the mean-based AND model. As shown in Figure 5.5, the linear-based model is almost the same as the first baseline model but replaces the mean layer with a fully connected layer, so the calculation of the joint representation $\mathbf{r}$ is:

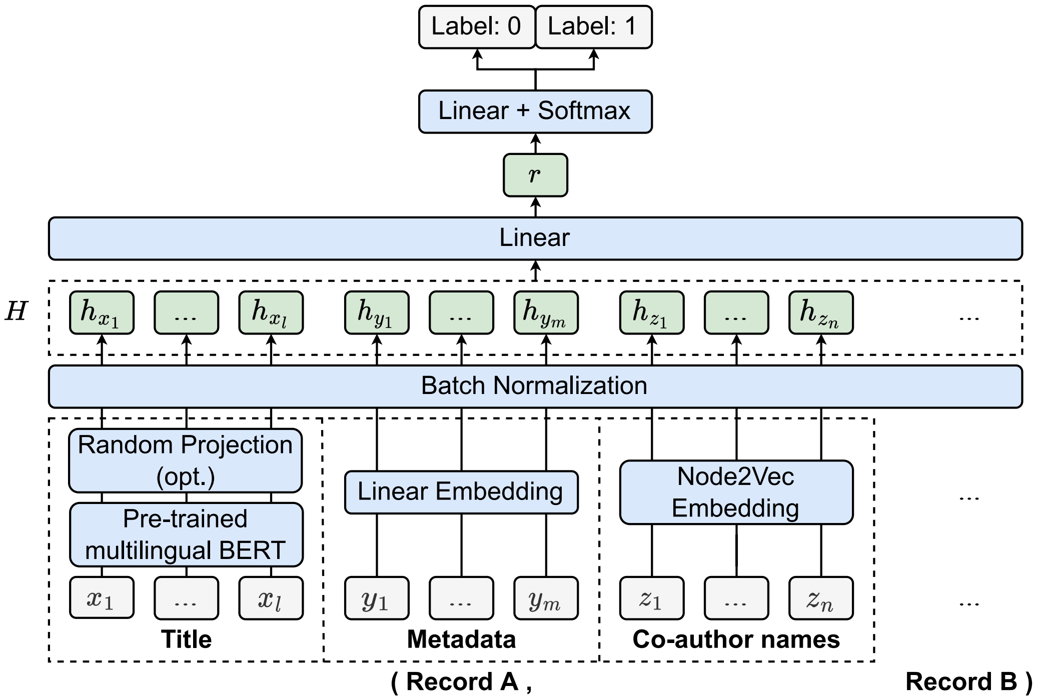


Figure 5.5: Baseline linear-based AND model.

$$\mathbf{r} = \mathbf{W}_r{}^T\mathbf{H} + \mathbf{b}_r, \tag{5.16}$$

where $\mathbf{W}_r \in \mathbb{R}^{2N\times 1}$and $\mathbf{b}_r \in \mathbb{R}^1$ are trainable parameters.

5.3 Baseline Text Models

Another baseline model is the text-only model, which only processes book titles to distinguish authorship. The entire structure of the text-only model is shown in Figure 5.6. We input title pairs into the text-only model, concatenate two titles, and process them using the pre-trained multilingual BERT. Next, the context vector for the two titles produced by the BERT is inputted to a fully connected layer to predict whether the record pair is positive or not. The random project layer is used to produce the same-dimensional embeddings as the co-attention model.

The last baseline model is the full-text model, which inputs all three kinds of features as strings. As shown in Figure 5.7, the full-text model connects all features in the two records and encodes them into tokens. Same as the title-only model, the full-text model processes tokens with pre-trained multilingual BERT and uses a fully connected layer to output results.

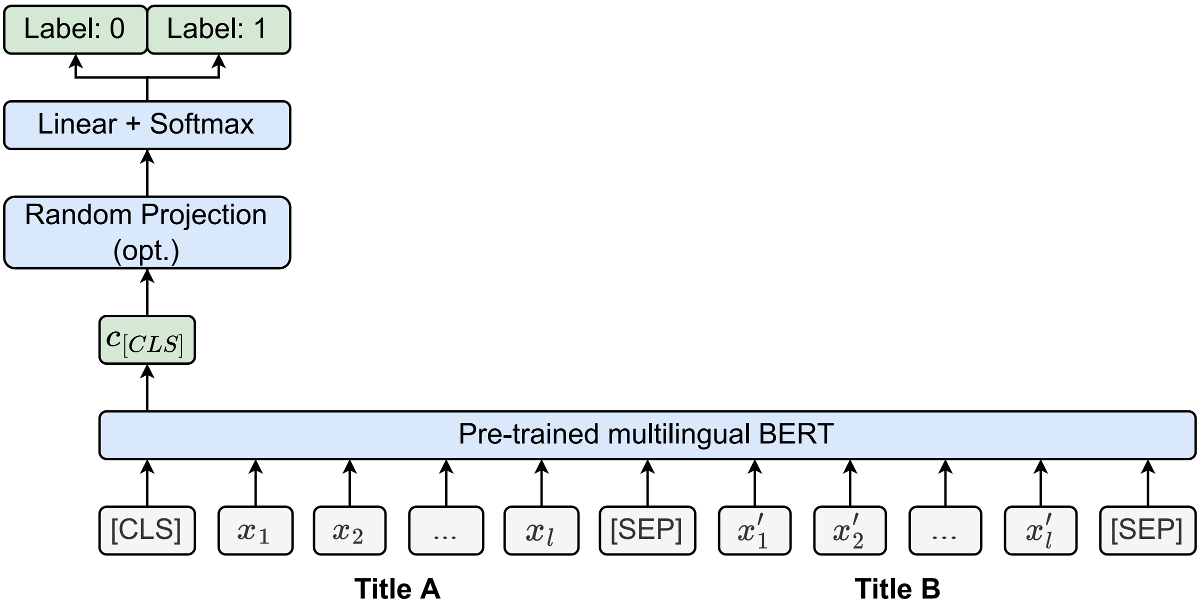


Figure 5.6: Baseline title-only model.

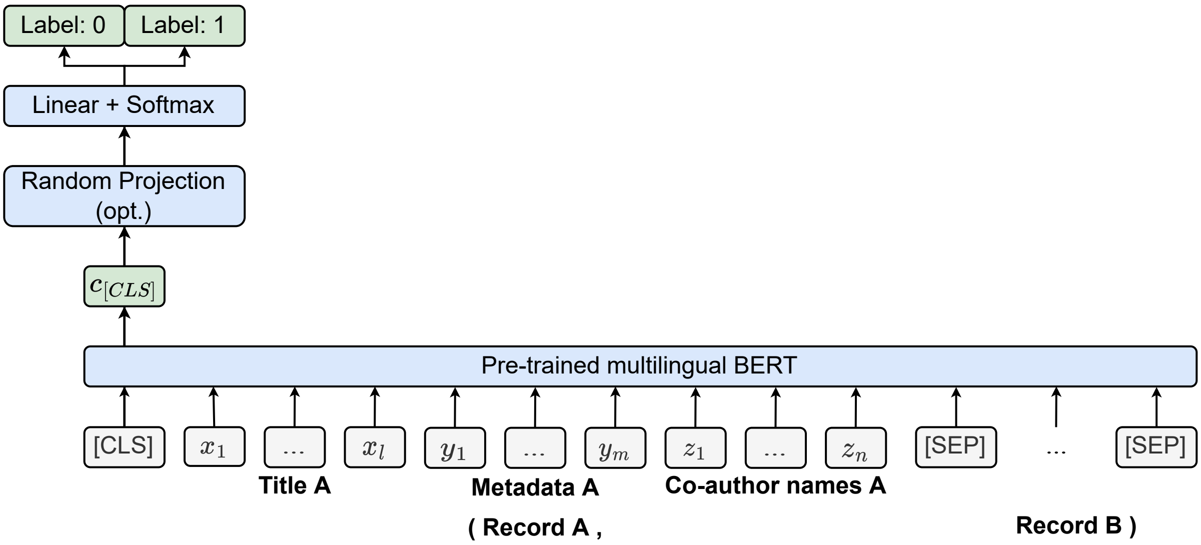


Figure 5.7: Baseline full-text model.

6 Experimentation

To answer the RQ1, we have to apply our proposed co-attention-based AND model and four baseline methods under the three datasets and compare their performances using evaluation metrics. Besides, we should visualise and analyse the attention weights of samples in the test set to answer the RQ2. For the RQ3 (field importance), we need to average attention weights on each field in the bibliographic data and compare the scores across all fields.

This chapter presents the experimental setups, including libraries and parameter settings in Section 6.1, and then introduces relevant evaluation metrics in Section 6.2.

Table 6.1: Learning rates of all models on the record-splitting datasets.

Model	Dataset	Learning Rate
Co-Attention-Based AND Model	Small	4.0E-04
	Medium	3.0E-04
	Large	6.0E-04
Baseline Mean-Based AND Model	Small	2.5E-02
	Medium	5.0E-03
	Large	1.5E-03
Baseline Linear-Based AND Model	Small	1.0E-02
	Medium	2.0E-03
	Large	7.0E-04
Baseline Title-Only 768 AND Model	Small	5.0E-03
	Medium	1.0E-03
	Large	2.0E-04
Baseline Title-Only 128 AND Model	Small	2.0E-04
	Medium	1.0E-04
	Large	3.0E-05
Baseline Full-Text AND Model	Small	1.0E-03
	Medium	6.0E-04
	Large	2.0E-04

6.1 Experimental Setup

Table 6.2: Learning rates of all models on the block-splitting datasets.

Model	Training Set	Learning Rate
Co-Attention-Based AND Model	Small	2.0E-03
	Medium	2.0E-04
	Large	4.0E-04
	Extra-Large	2.0E-04
Baseline Mean-Based AND Model	Small	2.0E-01
	Medium	2.0E-02
	Large	3.0E-03
	Extra-Large	6.0E-04
Baseline Linear-Based AND Model	Small	4.0E-02
	Medium	1.0E-02
	Large	2.0E-03
	Extra-Large	1.0E-03
Baseline Title-Only 128 AND Model	Small	1.0E-02
	Medium	4.0E-04
	Large	6.0E-05
	Extra-Large	5.0E-06
Baseline Full-Text AND Model	Small	1.0E-02
	Medium	4.0E-04
	Large	6.0E-05
	Extra-Large	5.0E-05

In the experiments, we apply the frozen BERT multilingual cased model [61] provided by the Huggingface' transformers library [83] to process record titles. The number of title tokens of each record is limited to up to 20. The number of tokens of each record in the full-text model is limited to up to 40. For coauthor information, we first construct an undirected coauthor network for each dataset using NetworkX [84], then apply *node2vec* [68] to generate node embeddings for all author names because the coauthor information is inputted to our model as initial-surnames. When generating node embeddings, we set the output dimension as 128, the walk length as 20, and the number of walks as 10. We only input at most five coauthors for each record to simplify the visualisation interface. Besides, we implement all models using the high-performance deep learning library PyTorch [85].

All models use Adam [86] as the optimiser, which is an efficient stochastic gradient descent method. We keep all parameters as default except the learning rate. The learning rates of all models on the three datasets are determined by the learning rate estimation method described in this paper [87]. Over an epoch, the model starts with a low enough learning rate (1E-8) and gradually increases it to a high learning rate (1E0 or even 1E1). After drawing the loss line against learning rates, we can choose the relatively better learning rate from a learning rate which is approaching the minimum loss on the left side of the minimum. All learning rates used are shown in Table 6.1 and Table 6.2 and the detailed learning rate finding results are shown in Appendix B and Appendix C.

6.2 Evaluation

We apply pairwise accuracy, F1 and Area Under the Receiver Operating Characteristic Curve (ROC AUC) to measure the performance of all methods, according to some previous AND algorithms [35, 88].

Let TP be the number of samples correctly labelled as positive, TN represent the number of samples correctly classified as negative, FP equal the number of samples incorrectly labelled as positive, and FN indicate the number of samples incorrectly classified as negative.

1. **Accuracy**: $\mathrm{Acc} = (\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})$,
2. **Precision**: $\mathrm{Prec} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$,
3. **Recall**: $\mathrm{Rec} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$,
4. **F1**: $\mathrm{F1} = (2 \times \mathrm{Prec} \times \mathrm{Rec})/(\mathrm{Prec} + \mathrm{Rec})$,
5. **true positive rate**: $\mathrm{TPR} = \mathrm{Rec} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$,
6. **false positive rate**: $\mathrm{FPR} = \mathrm{FP}/(\mathrm{FP} + \mathrm{TN})$.

Accuracy measures the overall number of correct predictions against the number of samples. F1 score is the harmonic mean of precision and recall. ROC AUC is measured using the area under the ROC curve, which is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings [89].

7 Results

In this chapter, the results of a series of experiments are reported. Based on the results on both the record-splitting and block-splitting datasets, our proposed method and four baseline methods are performed to evaluate their performance in AND tasks. Then, the decision evidence of the co-attention based model and the attribute importance results are also presented.

We apply the proposed method and three baseline models to the three AND datasets. Each dataset consists of record pairs and labels indicating the record pair is written by the same or different authors. Every record pair contains two records that are taken from the publications of the same name block. The co-attention-based AND model, baseline mean-based model, and baseline linear-based model all embed title tokens, tabular values, and coauthors into context vectors of dimension 128 and apply different fusion techniques to the context vectors. The title-only model is different and embeds the two titles of a record pair into a single context vector of dimension 128. Similarly, the full-text model takes all attributes as strings and embeds all tokens of a record pair into a single context vector of dimension 128.

Table 7.1: The detailed results (%) of all methods on the record-splitting datasets.

Model	**Small**			**Medium**			**Large**		
	Acc	**F1**	**ROC**	**Acc**	**F1**	**ROC**	**Acc**	**F1**	**ROC**
Co-Attn-Based AND	**83.05**	**83.10**	**90.24**	**86.24**	**86.38**	**93.15**	**87.62**	**87.73**	**94.16**
Linear-Based AND	75.78	77.42	83.88	78.02	78.96	85.93	83.64	83.61	90.99
Mean-Based AND	63.71	68.91	60.56	63.75	69.41	60.16	64.79	70.11	61.41
Title-Only AND	72.10	71.73	80.41	72.93	72.54	81.34	73.18	73.26	81.55
Full-Text AND	74.70	75.05	83.44	76.23	75.74	84.64	76.43	75.94	84.80

The classification results of the five models on the record-splitting datasets are shown in Table 7.1, and that of the five models on the block-splitting datasets are shown in Table 7.2. Because the validation and test sets are fixed in the block-splitting datasets, we can plot models' performance on different training set sizes as Figure 7.1.

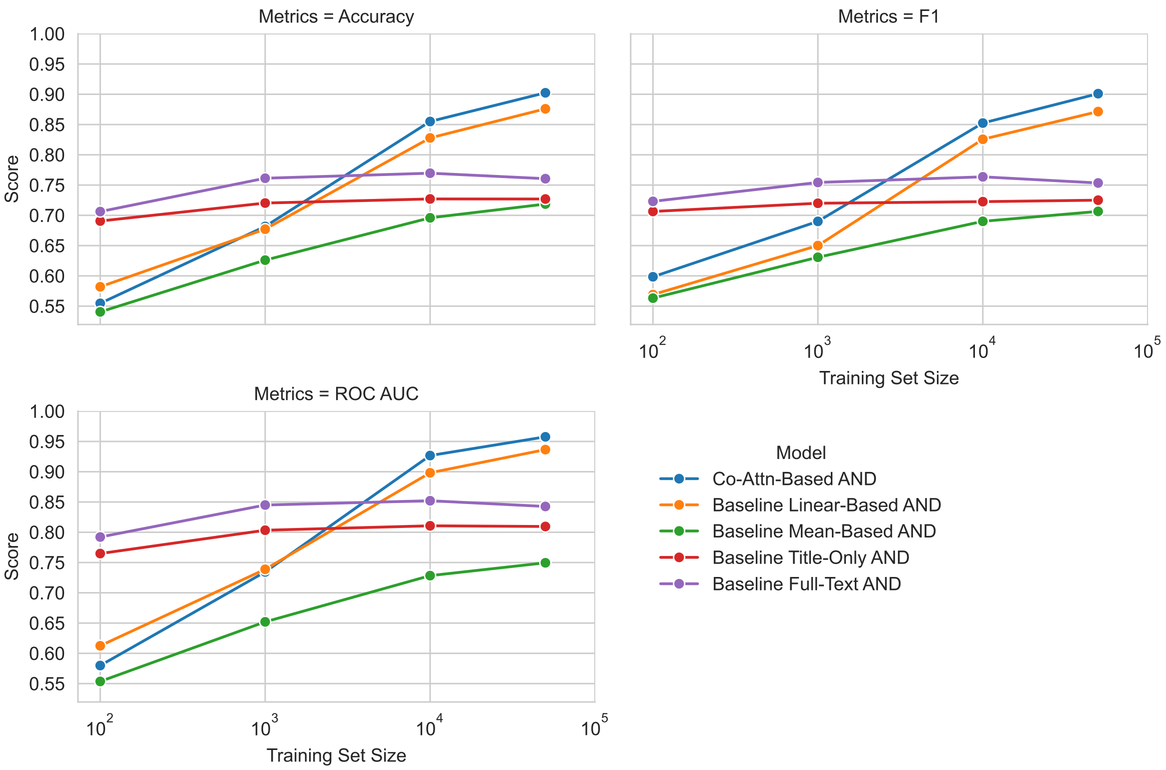


Figure 7.1: The results of models on the block-splitting datasets.

Table 7.2: The detailed results (%) of all methods on the block-splitting datasets.

Model	**Small**			**Medium**			**Large**			**Extra-Large**		
	Acc	**F1**	**ROC**	**Acc**	**F1**	**ROC**	**Acc**	**F1**	**ROC**	**Acc**	**F1**	**ROC**
Co-Attn-Based AND	55.46	59.85	57.99	68.17	69.00	73.45	**85.50**	**85.25**	**92.67**	**91.02**	**90.94**	**96.14**
Linear-Based AND	58.20	56.91	61.25	67.72	65.00	73.88	82.79	82.57	89.82	86.74	86.00	93.11
Mean-Based AND	54.05	56.32	55.36	62.60	63.07	65.20	69.59	69.01	72.83	71.63	70.31	75.00
Title-Only AND	69.06	70.64	76.48	72.03	72.00	80.34	72.71	72.26	81.07	72.55	72.33	81.01
Full-Text AND	**70.62**	**72.31**	**79.21**	**76.13**	**75.43**	**84.50**	76.96	76.36	85.21	76.21	75.22	84.48

7.1 Results of Baseline Text Models

Compared with the baseline title-only model, the full-text model, inputs all attributes as strings, is about 1-4% better in the accuracy score, 2-4% better in the F1 score, and 3-4% better in the ROC AUC score on both the record-splitting and block-splitting datasets. The dataset sizes only have limited effect on the two baseline text models that the baseline text models are about 1% better using the largest dataset than that using the smallest dataset.

To evaluate the impact of random projection, we compare the baseline title-only model with and without random projection on all three record-splitting datasets and present the results in Table 7.3. Based on the accuracy, F1 and ROC AUC scores, the random projection worsens the model's performance by about 1-2% in the three datasets. In addition, we also compared the running time difference of the title-only model with and without random projection in the record-splitting small dataset, as shown in Table 7.4. Although we need 8 minutes and 15 seconds to generate and cache BERT outputs, the training time of each epoch is dramatically reduced from 15 minutes and 52 seconds to 18 seconds. Overall, with the help of random projection, we can save lots of running time while just sacrificing about 1% performance.

Table 7.3: The results (%) of the two title-only models on the record-splitting datasets.

Model	Small			Medium			Large		
	Acc	**F1**	**ROC**	**Acc**	**F1**	**ROC**	**Acc**	**F1**	**ROC**
Text-Only AND (768)	73.45	73.48	82.07	73.75	74.06	82.35	74.26	74.20	82.77
Text-Only AND (128)	72.10	71.73	80.41	72.93	72.54	81.34	73.18	73.26	81.55

Table 7.4: The running time of the two title-only models on the record-splitting small dataset.

	Generating BERT Outputs	**Training Time of One Epoch**
Title-Only AND (128)	8m15s	18s
Title-Only AND (768)	None	15m52s

7.2 Results of Baseline All-Attribute Models

Under the record-splitting datasets, the mean-based model is always the worst. Compared with the title-only model, the mean-based model is about 9.4% worse in accuracy, 3% worse in F1 score, and 20% worse in ROC AUC score in all three datasets. Although the mean-based model is still the worst in all four training sets in the block-splitting datasets, its performance increases slowly with the increment of training set size. The mean-based model performs poorly mainly because the network structure is too simple to handle various kinds of data of two records together, and the mean layer is insufficient for the joint representation of all context vectors.

However, the linear-based AND model is about 3.5-10% better in accuracy, 6-10% in F1 score and 3.5-10% in ROC AUC score in the three record-splitting datasets compared with the baseline text-only model. Besides, the performance of the linear-based AND model in the block-splitting datasets soars quickly with the increment of training set size and outperform the two text models in the large and 5xlarge datasets. In conclusion, the additional metadata and coauthor information help the model classify record pairs in AND tasks.

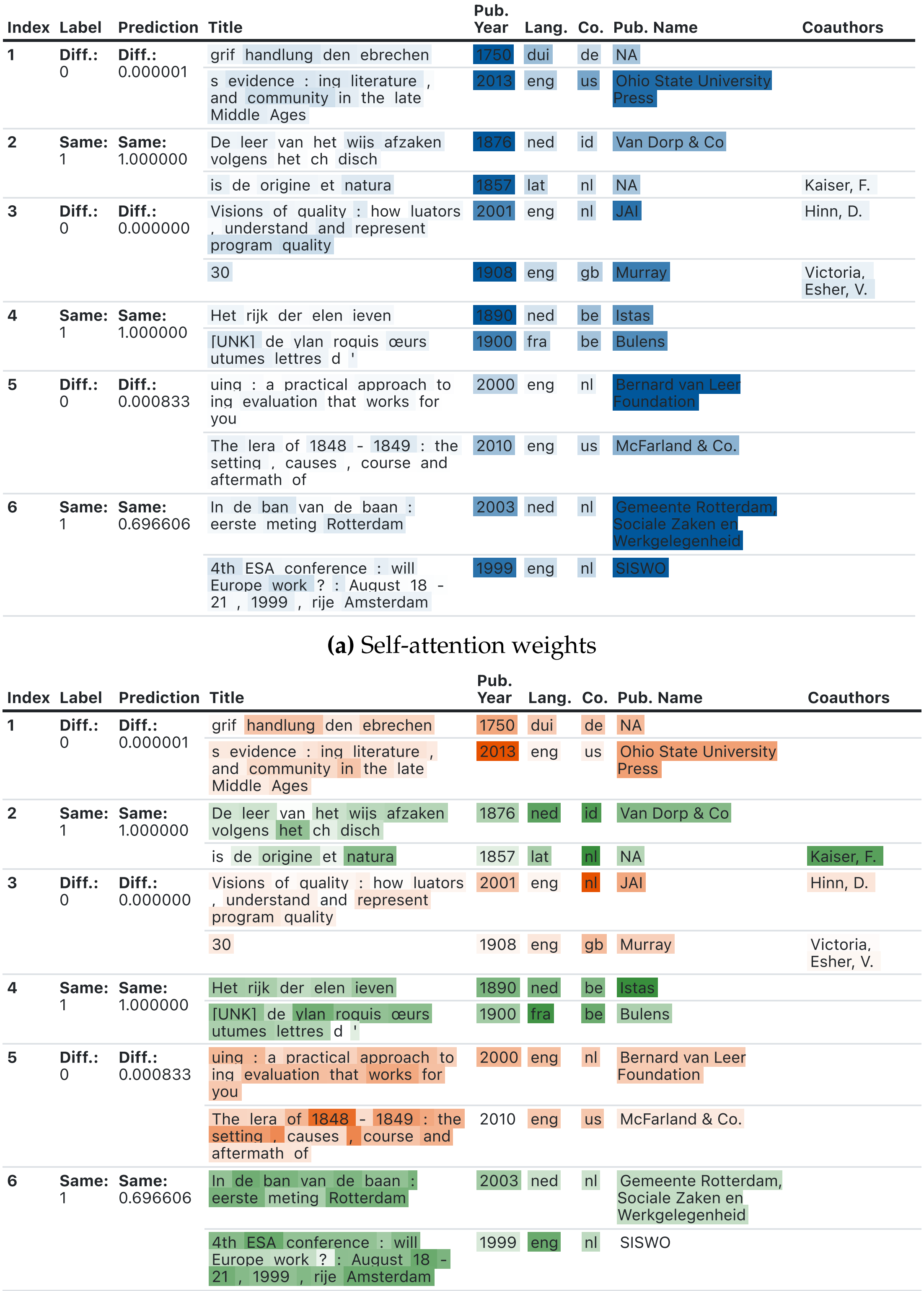
7.3 Results of The Co-Attention-Based Model

The co-attention-based AND model is the best in all three record-splitting datasets according to Table 7.1. It is about 4-8% better in accuracy, 4-8% in F1 score, and 3-7% in ROC AUC than the baseline linear-based AND model. When using the block-splitting datasets, the co-attention-based model performs badly when the dataset is too small as the two baseline full-attribute models. Nevertheless, in most cases, the co-attention-based model is still better than the baseline full-attribute models, which means the attention mechanisms detect authorships of record pairs better than the fully connected layer.

7.3.1 Evidence of Model Decisions

Another experiment is to visualise attention weights and provide evidence of how the co-attention-based model predicts authorship relations of record pairs. It explores whether the self-attention and co-attention mechanisms are suitable for annotating crucial components in record pairs that activate predictions. The original attention weights are calculated by softmax. It results in that some records have attention weights less than 0.1, which cannot be shown properly in a colour system. In order to better present the relative importance of different elements, we transform the attention weights to a fixed range $(0, 1)$. Therefore, elements with attention weight 1 are shown using the darkest colour, while elements with attention weight 0 are shown using white.

Here we take the co-attention-based model in the record-splitting large dataset as an example. The visualisation of self-attention and co-attention weights of samples in the large dataset is presented in Figure 7.2. We display some record pairs with obvious attention weights as tables. In the table, every two rows constitute a record pair written by authors with the same name. The first two columns present the true labels and the model's predictions. Other columns are record attributes containing titles, publication years, languages, countries, publisher names, and coauthors. The number of title tokens and coauthors is flexible.



Index	Label	Prediction	Title	Pub. Year	Lang.	Co.	Pub. Name	Coauthors
1	Diff.: 0	Diff.: 0.000001	grif handlung den ebrechen	1750	dui	de	NA	
			s evidence : ing literature , and community in the late Middle Ages	2013	eng	us	Ohio State University Press	
2	Same: 1	Same: 1.000000	De leer van het wijs afzaken volgens het ch disch	1876	ned	id	Van Dorp & Co	
			is de origine et natura	1857	lat	nl	NA	Kaiser, F.
3	Diff.: 0	Diff.: 0.000000	Visions of quality : how luators , understand and represent program quality	2001	eng	nl	JAI	Hinn, D.
			30	1908	eng	gb	Murray	Victoria, Esher, V.
4	Same: 1	Same: 1.000000	Het rijk der elen ieven	1890	ned	be	Istas	
			[UNK] de vlan roquis œurs utumes lettres d '	1900	fra	be	Bulens	
5	Diff.: 0	Diff.: 0.000833	uing : a practical approach to ing evaluation that works for you	2000	eng	nl	Bernard van Leer Foundation	
			The lera of 1848 - 1849 : the setting , causes , course and aftermath of	2010	eng	us	McFarland & Co.	
6	Same: 1	Same: 0.696606	In de ban van de baan : eerste meting Rotterdam	2003	ned	nl	Gemeente Rotterdam, Sociale Zaken en Werkgelegenheid	
			4th ESA conference : will Europe work ? : August 18 - 21 , 1999 , rije Amsterdam	1999	eng	nl	SISWO	

(a) Self-attention weights

Index	Label	Prediction	Title	Pub. Year	Lang.	Co.	Pub. Name	Coauthors
1	Diff.: 0	Diff.: 0.000001	grif handlung den ebrechen	1750	dui	de	NA	
			s evidence : ing literature , and community in the late Middle Ages	2013	eng	us	Ohio State University Press	
2	Same: 1	Same: 1.000000	De leer van het wijs afzaken volgens het ch disch	1876	ned	id	Van Dorp & Co	
			is de origine et natura	1857	lat	nl	NA	Kaiser, F.
3	Diff.: 0	Diff.: 0.000000	Visions of quality : how luators , understand and represent program quality	2001	eng	nl	JAI	Hinn, D.
			30	1908	eng	gb	Murray	Victoria, Esher, V.
4	Same: 1	Same: 1.000000	Het rijk der elen ieven	1890	ned	be	Istas	
			[UNK] de vlan roquis œurs utumes lettres d '	1900	fra	be	Bulens	
5	Diff.: 0	Diff.: 0.000833	uing : a practical approach to ing evaluation that works for you	2000	eng	nl	Bernard van Leer Foundation	
			The lera of 1848 - 1849 : the setting , causes , course and aftermath of	2010	eng	us	McFarland & Co.	
6	Same: 1	Same: 0.696606	In de ban van de baan : eerste meting Rotterdam	2003	ned	nl	Gemeente Rotterdam, Sociale Zaken en Werkgelegenheid	
			4th ESA conference : will Europe work ? : August 18 - 21 , 1999 , rije Amsterdam	1999	eng	nl	SISWO	

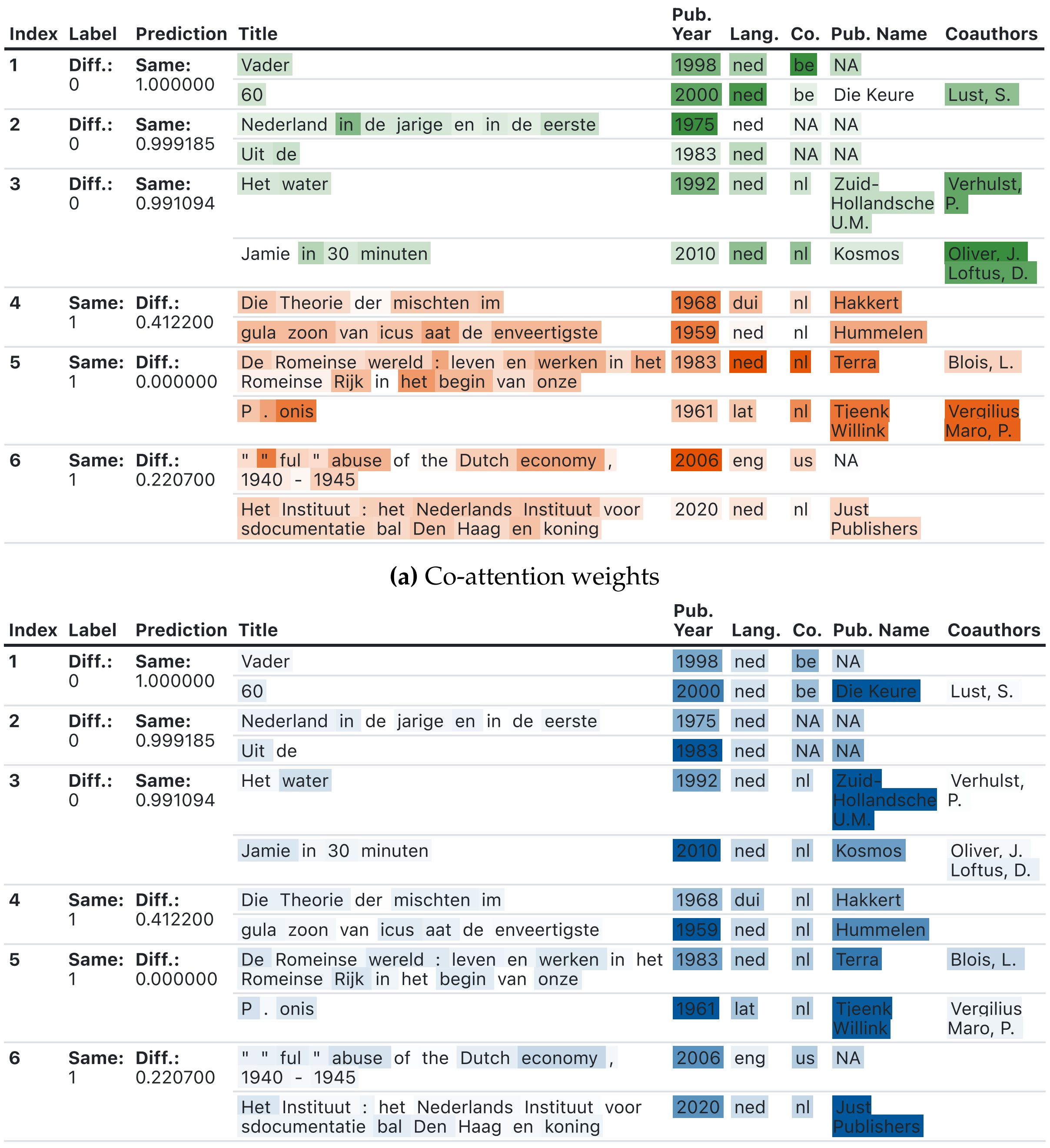
(b) Co-attention weights

Figure 7.2: Attention weights of correctly classified examples on the large dataset.

In Figure 7.2a, blue colours indicate important elements for the self-attention module, and a darker colour indicates greater importance. In the first sample, the two records are written by different authors. It can be easily classified because the two records have titles on different topics, hugely different publication years, and different languages, countries, and publishers. The self-attention mechanism focuses most on the publication years ″1750″ and ″2013″ and the publisher name ″Ohio State University Press″. Meanwhile, their languages, country codes, and the title words ″handlung″, ″evidence″, and ″community″ also gain some attention.

Next, we explore the co-attention weights generated by the co-attention module. As shown in Figure 7.2b, the green items make the model classify the two records as a positive pair, while the red items work oppositely. Similar to the self-attention module, darker items are more influential than lighter items. For the second and third samples in Figure 7.2a, the self-attention mechanism puts its focus primarily on publication years and publisher names, and only a little attention is on titles. In contrast, the co-attention weights of the two samples were distributed in different features. The languages, countries, coauthors, and several title words also get attention and cannot be ignored. In addition, The co-attention mechanism gives more attention to meaningful words than the self-attention mechanism. For example, in the sixth sample, the self-attention mechanism focuses on the words ″ban″, ″Rotterdam″, ″4th″, ″work″, and ″1999″ while the words ″eerste″, ″ESA″, and ″Amsterdam″ are also captured by the co-attention mechanism.

To analyse the interpretability comprehensively, we also need to check the misclassified samples. As shown in Figure 7.3, the green or red colours are based on predictions. The first two samples suffer from short titles, so the model cannot get enough latent information from titles. Meanwhile, their records have close publication years and the same languages and countries. The third sample has distant publication years and different publisher names and coauthors. It tends to be written by different authors, but the model predicts it as the ″same″. In addition, the last three samples in Figure 7.3 are misclassified as negative samples. They all have different metadata and coauthors. The two records in the fourth sample are on different topics, and the fifth sample suffers from short titles as its second title has no meaning at all. These errors make sense because even humans cannot predict them easily.



Index	Label	Prediction	Title	Pub. Year	Lang.	Co.	Pub. Name	Coauthors
1	Diff.: 0	Same: 1.000000	Vader	1998	ned	be	NA	
			60	2000	ned	be	Die Keure	Lust, S.
2	Diff.: 0	Same: 0.999185	Nederland in de jarige en in de eerste	1975	ned	NA	NA	
			Uit de	1983	ned	NA	NA	
3	Diff.: 0	Same: 0.991094	Het water	1992	ned	nl	Zuid-Hollandsche U.M.	Verhulst, P.
			Jamie in 30 minuten	2010	ned	nl	Kosmos	Oliver, J. Loftus, D.
4	Same: 1	Diff.: 0.412200	Die Theorie der mischten im	1968	dui	nl	Hakkert	
			gula zoon van icus aat de enveertigste	1959	ned	nl	Hummelen	
5	Same: 1	Diff.: 0.000000	De Romeinse wereld : leven en werken in het Romeinse Rijk in het begin van onze	1983	ned	nl	Terra	Blois, L.
			P . onis	1961	lat	nl	Tjeenk Willink	Vergilius Maro, P.
6	Same: 1	Diff.: 0.220700	" " ful " abuse of the Dutch economy , 1940 - 1945	2006	eng	us	NA	
			Het Instituut : het Nederlands Instituut voor sdocumentatie bal Den Haag en koning	2020	ned	nl	Just Publishers	

(a) Co-attention weights

Index	Label	Prediction	Title	Pub. Year	Lang.	Co.	Pub. Name	Coauthors
1	Diff.: 0	Same: 1.000000	Vader	1998	ned	be	NA	
			60	2000	ned	be	Die Keure	Lust, S.
2	Diff.: 0	Same: 0.999185	Nederland in de jarige en in de eerste	1975	ned	NA	NA	
			Uit de	1983	ned	NA	NA	
3	Diff.: 0	Same: 0.991094	Het water	1992	ned	nl	Zuid-Hollandsche U.M.	Verhulst, P.
			Jamie in 30 minuten	2010	ned	nl	Kosmos	Oliver, J. Loftus, D.
4	Same: 1	Diff.: 0.412200	Die Theorie der mischten im	1968	dui	nl	Hakkert	
			gula zoon van icus aat de enveertigste	1959	ned	nl	Hummelen	
5	Same: 1	Diff.: 0.000000	De Romeinse wereld : leven en werken in het Romeinse Rijk in het begin van onze	1983	ned	nl	Terra	Blois, L.
			P . onis	1961	lat	nl	Tjeenk Willink	Vergilius Maro, P.
6	Same: 1	Diff.: 0.220700	" " ful " abuse of the Dutch economy , 1940 - 1945	2006	eng	us	NA	
			Het Instituut : het Nederlands Instituut voor sdocumentatie bal Den Haag en koning	2020	ned	nl	Just Publishers	

(b) Self-attention weights

Figure 7.3: Attention weights of misclassified samples in the large dataset.

7.3.2 Attribute Importance

The third sub-question aims to investigate the importance of each attribute according to attention weights in the co-attention-based AND model. In order to measure the attribute importance rather than token importance in Section 7.3.1, we sum the attention weights of tokens belonging to the same attribute in each record as the weight of that attribute in the record. For example, the attention weight of a single title is calculated by

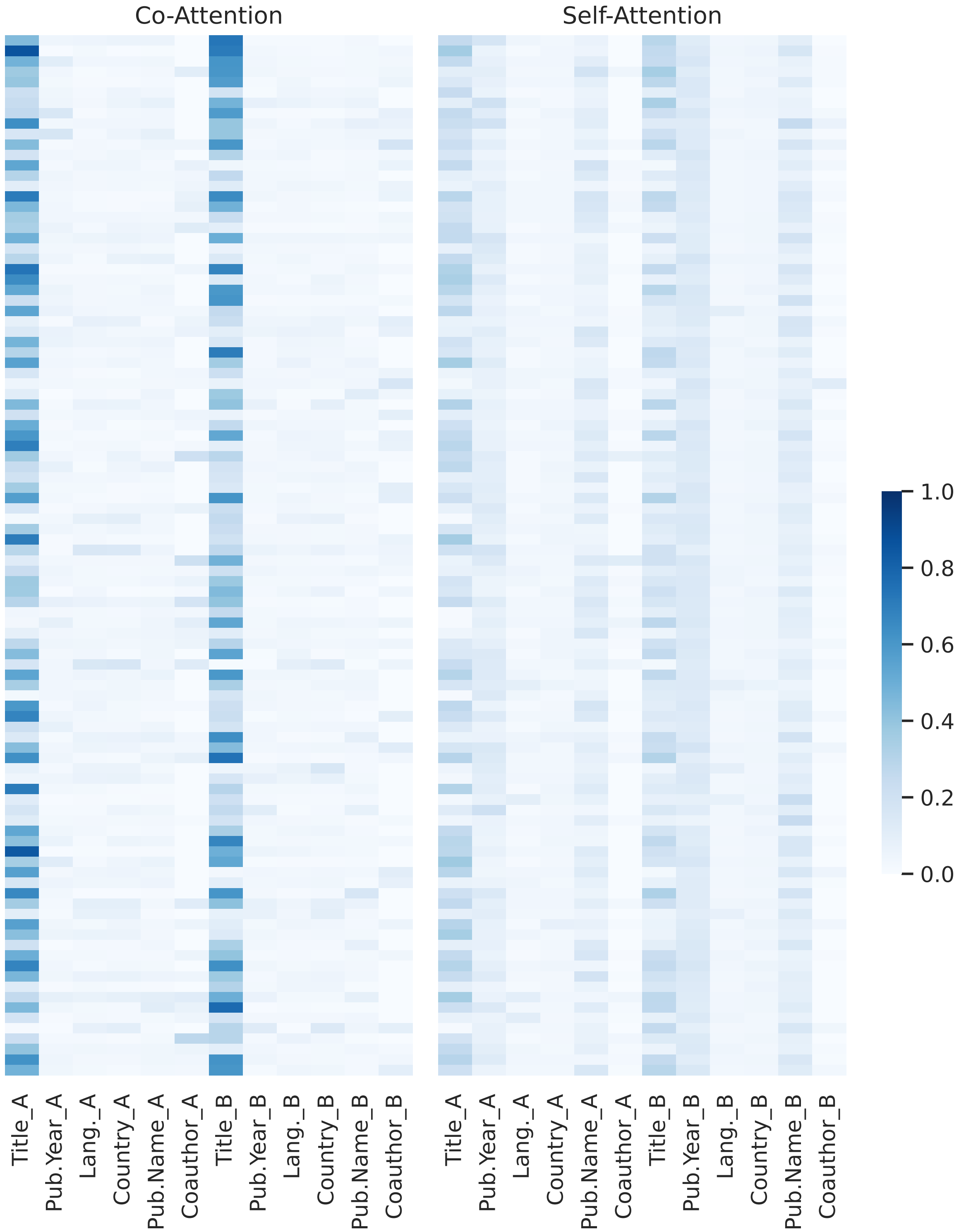


Figure 7.4: The results of attribute importance on randomly sampled 100 record pairs of the test set in the record-splitting large dataset.

summing the weights of all tokens of that title. Similarly, a distinct coauthor feature's attention weight is the summation of all coauthors' weights.

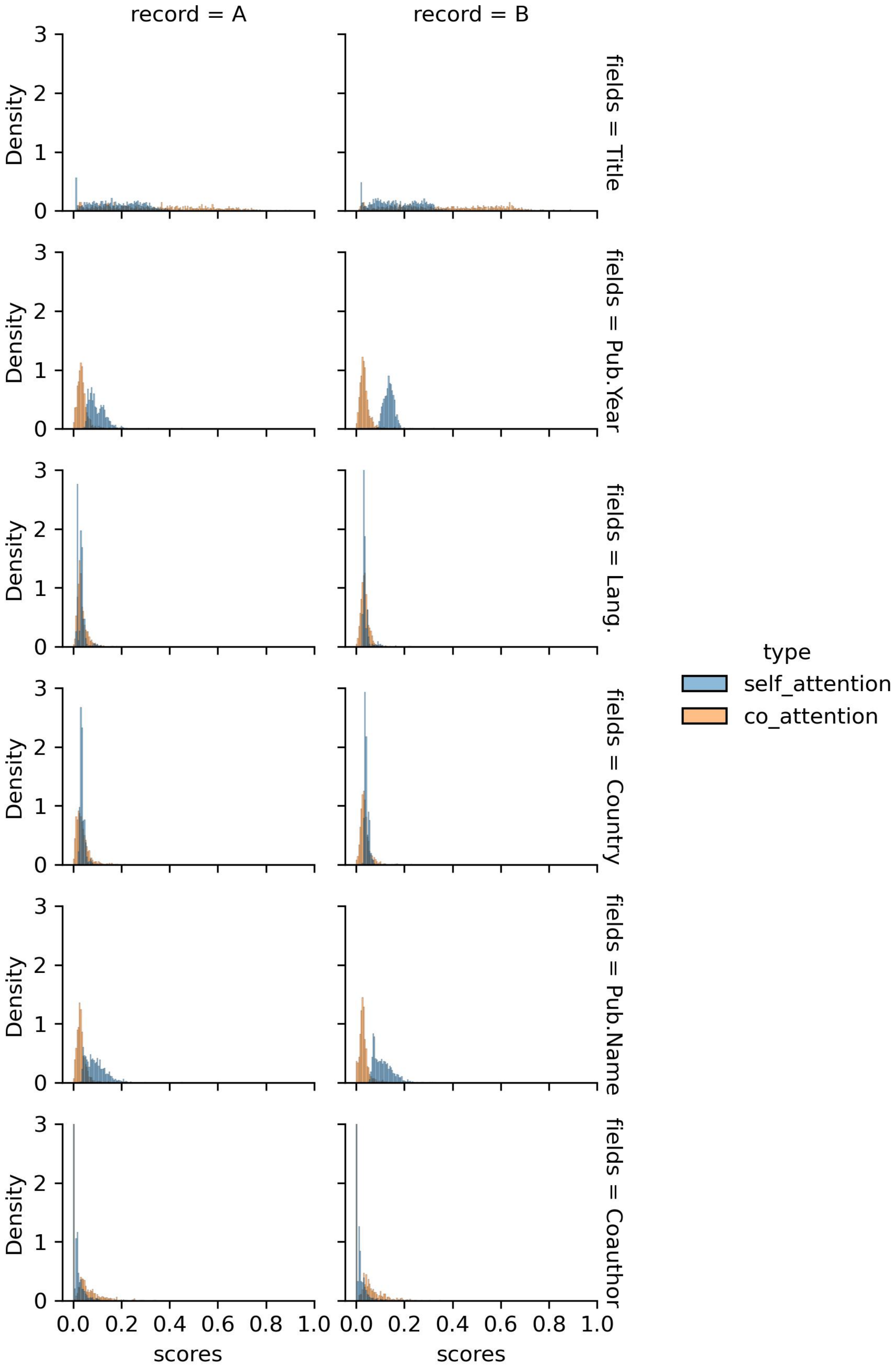


Figure 7.5: The distribution of attention weights of each field in the record-splitting large dataset.

As shown in Figure 7.4, the attribute importance varies in different records. In most records, titles have darker colours than other features according to both the co-attention and self-attention weights. By analysing the distribution of attention weights in each field in Figure 7.5 and the statistical description of attention weights of each feature in Table 7.5, titles are the most important feature for the co-attention mechanism, with the mean co-attention weight of 0.3402. For the self-attention mechanism, titles and publication years are more important than other features, with mean weights of 0.1697 and 0.1161, respectively.

Table 7.5: The statistical description of attention weights of each attribute on the record-splitting large dataset.

Stat	**Co-Attention Weights**						**Self-Attention Weights**					
	Title	**Pub. Year**	**Lang.**	**Co.**	**Pub. Name**	**Coau.**	**Title**	**Pub. Year**	**Lang.**	**Co.**	**Pub. Name**	**Coau.**
count	4,041,532						4,041,532					
mean	0.3402	0.0349	0.0352	0.0362	0.0322	0.0388	0.1697	0.1161	0.0346	0.0399	0.1034	0.0146
std	0.2102	0.0217	0.0191	0.0229	0.0213	0.0521	0.0971	0.0330	0.0158	0.0095	0.0422	0.0202
min	0.0009	0.0000	0.0000	0.0000	0.0000	0.0000	0.0090	0.0287	0.0099	0.0164	0.0186	0.0000
25%	0.1618	0.0218	0.0237	0.0222	0.0203	0.0000	0.0861	0.0892	0.0300	0.0338	0.0725	0.0000
50%	0.3150	0.0316	0.0318	0.0319	0.0289	0.0243	0.1649	0.1208	0.0332	0.0379	0.0966	0.0105
75%	0.5176	0.0425	0.0422	0.0438	0.0386	0.0599	0.2538	0.1403	0.0364	0.0432	0.1293	0.0201
max	0.9998	0.7446	0.9630	0.5047	0.8196	0.9099	0.5661	0.2682	0.2216	0.1291	0.4146	0.2185

Note: Pub.Year represents the year of publication, Lang. represents the language of the publication, Co. represents the publication country, Pub.Name is the publisher name, and Coau. represents the coauthor.

8 Discussion

This chapter elaborates critical discussion about our study. The answers and relevant discussion of each research question are presented in the first three sections. The following section introduces limitations found in the methods of AND and outlines possible ideas for future work.

8.1 RQ1: Impact of Multiple Attributes

The RQ1 is about the improvements for AND tasks with more attributes other than only using titles. Titles are usually short summaries and contain semantic information about publications. With the help of latent information involved in the pre-trained multilingual BERT model, the baseline title-only AND model can simultaneously process titles of two records and investigate potential connections between two titles. The performance of the baseline title-only model on all three record-splitting datasets is identically high without any fine-tuning. It achieves 72% for accuracy and F1 and 81% for ROC AUC in all three different-sized datasets. Besides, its performance on block-splitting datasets ranges from 69-81%. The results are quite acceptable as only an additional linear layer is trainable in the title-only model.

Based on this title-only model and other accessible attributes, such as publication year, language, etc., we proposed the baseline full-text model to verify the impact of multiple attributes as strings. The additional attributes improve the text model by about 2-4%, which means more information helps the model make better predictions.

8.2 RQ2: Impact of Multiple Kinds of Features with Corresponding Techniques

We proposed two baseline full-attributes models for AND, the linear-based model and the mean-based model. Among the three datasets, the mean-based model is insufficient to understand record similarity and can not classify record pairs well, compared with the baseline text-only model. An intuitive explanation is that the mean-based model loses too much meaningful information when averaging the hidden context vectors generated from titles, metadata, and coauthor information. It also indicates that the importance of each context vector differs considerably that an unweighted mean suffers from extracting and summarizing enough information for further classification.

The baseline linear-based AND model utilizes a fully connected layer to incorporate all hidden context vectors into a joint representation. The linear-based model performs much better than the mean-based model as it applies a linear function to convert vectors. Compared with the full-text model, the linear-based model outperforms by 1-8% according to the metrics scores under the record-splitting datasets, with the extra information from metadata and coauthor information.

When it comes to the block-splitting datasets, the linear-based model is not better than the text models on all occasions but starts at low scores and increases the performance with the increment of training set size. In other words, the linear-based model is hard to learn enough knowledge from a too-small training set which does not have a tight relationship with the validation and test set.

8.3 RQ3: Impact of Attention Mechanisms

The RQ1 concerns the improvements for AND models using the self-attention and co-attention mechanisms. It can be answered by analysing three sub-questions (a) the improvements in the model's performance made by attention weights, (b) evidence of model decisions, and (c) attribute importance, respectively.

8.3.1 RQ3.a: Improvements in Performance by Attention Weights

The linear-based AND model benefits from multiple feature classes and gains more helpful information from metadata and the coauthor graph. However, the fully connected layer in the linear-based method is non-transparent and cannot provide sufficient evidence for how it makes predictions. Besides, two records have no interactivities because the fully connected layer does not abstract records with their pairing record. All context information from two records is directly zipped into a single vector. The two limitations may affect its performance.

The proposed co-attention-based method outperforms by 3-8% in three different-sized record-splitting datasets with all three metrics than the linear-based method. Under the block-splitting datasets, the co-attention-based AND model also becomes better along with the training set size and is better than the linear-based model in most cases. The self-attention and co-attention mechanisms reasonably summarize the importance of each element for classification. With the collaboration of self-attention and co-attention, the model firstly forces elements of different kinds of attributes inside each record to interact with each other, then generates weights for elements in one record according to the elements in the other record. This way, the attention weights are interactive between not only attributes but also records.

8.3.2 RQ3.b: Evidence of Model Decisions

Attention mechanisms are usually straightforward to understand as they provide a discrete weight for each input item. Thus, the importance of title tokens, metadata features, and coauthors can be directly displayed using a table-structured interface, where each row consists of a record's feature, and every two rows compose a record pair, which is also a sample in datasets. The true labels and predictions of record pairs are given to determine whether the two records are positive (written by the same author) or negative (written by different authors). The colour depth indicates degrees of attention weights in the self-attention or co-attention modules.

By visualising the attention weights of items in different colour depths, the co-attention-based AND model provides evidence of model decisions to some extent. The self-attention weights indicate influential elements by investigating relationships between different attributes when only the current record information is provided. The co-attention module works differently. The co-attention weights are calculated by the elements of one record and all elements from another. The co-attention mechanism aims to find connections between the two records through the scaled dot-product attention method and provide evidence of why the two records are written by the same author and why not. Overall, Although the model's attention weights may not be identical to our expectations, the self-attention and co-attention mechanisms can indeed provide reasonable evidence for us when distinguishing authorships of a record pair.

8.3.3 RQ3.c: Attribute Importance

As the information of the [CLS] tokens comes from all other elements, if the model gives an element a higher attention weight, the information of that element is more retained. Therefore, the attribute importance can be presented by attention weights. Based on the self-attention and co-attention weights of items in every record, we can summarise feature importance by averaging the weights of every feature. Overall, the title feature is the most crucial according to the attention weights. The results are reasonable because titles typically contain rich semantic information and cover the main topics of publications. Besides, the other five attributes have equal importance in the co-attention mechanism, and the coauthor information ranks the last in the self-attention mechanism. Although the coauthor information is usually not ignorable when it exists in a record pair, there are no coauthors in many records. Lastly, the attention weights of features vary a lot in different samples because the attention mechanisms can focus on different components in different record pairs.

8.4 Limitations and Opportunities for Future Research

8.4.1 Feature Selection

Although we utilise three types of features in the proposed model, we only selected a limited subset of features that are recognised as necessary fields by the OCLC database specialists. Moreover, features with plenty of missing values are also ignored because dealing with missing values is not our primary interest in this thesis. For selected features, we retain the missing values, which can be better preprocessed with imputation methods or machine learning filling algorithms.

In addition, The construction approach of name blocks is not perfect because we merge authors by family name and the initial of the first name. Different from general text, personal names often have multiple valid variations. Besides, nicknames are widespread in daily life. Furthermore, personal names from different cultural backgrounds make the problem more serious. Thus, future research and applications can apply state-of-art personal name matching techniques to get better name blocks.

8.4.2 Embeddings

Due to the limitation on computing resources, we applied the pre-trained multilingual cased BERT and random projection as our processing module for textual features. The pre-trained BERT is frozen in our experiments to accelerate training. The random projection is used to reduce the dimension of word embeddings so that we can cache the word embeddings for reusability. Although the pre-trained model gets high metric scores in the AND tasks, and the random projection only worsens the performance by about 1%, we can use a better embeddings strategy in the future. For example, fine-tuning of state-of-art PTMs in OCLC's database is considerable. Fast and discriminative semantic embedding [90] can be applied if pre-training a language model with billions of parameters is unrealistic in applications.

We also consider using pre-trained methods for our tabular metadata to replace the simple linear embeddings. The pre-training of neural networks for tabular data is still a challenge. Moreover, a better method to generate node embeddings for coauthor networks might be useful.

8.4.3 Joint Representation and Interpretation

In the proposed co-attention model, we firstly investigate self-attentive connections between attributes inside a record, then calculate latent connections between two records using the co-attention mechanism. After that, each record's final representations with interactive information are generated by the co-attention mechanism using the latent connections and the context vectors. The attention mechanisms can interpret model decisions by visualising both the self-attention and co-attention weights. However, it is unknown to what extent these explanations can improve the manual labelling process.

In future work, a quantitative analysis might be helpful to measure the improvements made by attention mechanisms. In addition, more complex attention techniques such as transformers can be applied to investigate interactive connections between records. Moreover, it may be helpful to study how to extract and process attention weights of multi-layer and multi-head attention mechanisms as interpretations of classifications.

9 Conclusion

This thesis introduces the application of embedding techniques and attention mechanisms to the problem of pairwise author name disambiguation. Based on the results of the literature review, three significant limitations are found in previous works: (1) lack of state-of-art embedding techniques in feature processing, (2) only one or two types of features are used, and (3) no interpretation of model decisions. By processing each type of features with appropriate methods and combining both the self-attention and co-attention mechanisms to integrate hidden context vectors from all attributes of record pairs, we propose the co-attention-based AND model and compare it with four baseline methods in different-sized datasets with different splitting methods.

The results of experiments indicate that the additional attributes are helpful for AND tasks, and the self-attention and co-attention mechanisms give the model more opportunity to learn latent information behind the data. The integration of full attributes and attention mechanisms helps the model classify record pairs more precisely from the same name block with positive and negative labels. In addition, the visualisation of attention weights gives us a perception of how the model predicts record pairs. Although it is still debatable whether the attention weights are explanations or not, they certainly provide some intuitive evidence of the model's decision processes. Furthermore, according to the overall attention weights of each attribute, feature importance is concluded by statistical descriptions and also visualising the overall attention weights of randomly selected 100 samples. It shows that the title is the most important feature.

Overall, the co-attention-based AND model is a valuable strategy in addressing the challenges of multiple types of features processing and prediction interpretations in AND tasks. However, it still faces the problem of insufficient metadata preprocessing, fixed embedding techniques, and human-understanding difficulty. Therefore, OCLC can limit decision thresholds to get more precise classifications and use it as an assistant tool for manual labelling. Moreover, future work can be done to improve the AND models by constructing more accurate and representative datasets and investigating interpretation mechanisms that approach more comprehensive human understanding.

Bibliography

[1] Amy H Turner. Oclc worldcat as a cooperative catalog. *Cataloging & Classification Quarterly*, 48(2-3):271–278, 2010.

[2] Laurel L Haak, Martin Fenner, Laura Paglione, Ed Pentz, and Howard Ratner. Orcid: a system to uniquely identify researchers. *Learned publishing*, 25(4):259–264, 2012.

[3] Daniel Torres-Salinas, Wenceslao Arroyo-Machado, and Mike Thelwall. Exploring worldcat identities as an altmetric information source: A library catalog analysis experiment in the field of scientometrics. *Scientometrics*, 126(2):1725–1743, 2021.

[4] Anderson A. Ferreira, Marcos André Gonçalves, and Alberto H.F. Laender. A brief survey of automatic methods for author name disambiguation. *SIGMOD Rec.*, 41(2):15–26, aug 2012. ISSN 0163-5808. doi: 10.1145/2350036.2350040. URL `https://doi-org.ezproxy2.utwente.nl/10.1145/2350036.2350040`.

[5] Alexander Tekles and Lutz Bornmann. Author name disambiguation of bibliometric data: A comparison of several unsupervised approaches1. *Quantitative Science Studies*, 1(4):1510–1528, 12 2020. ISSN 2641-3337. doi: 10.1162/qss_a_00081. URL `https://doi.org/10.1162/qss\_a\_00081`.

[6] Xiaoming Fan, Jianyong Wang, Xu Pu, Lizhu Zhou, and Bing Lv. On graph-based name disambiguation. *Journal of Data and Information Quality (JDIQ)*, 2(2):1–23, 2011.

[7] Jun Xu, Siqi Shen, Dongsheng Li, and Yongquan Fu. A network-embedding based method for author disambiguation. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 1735–1738, 2018.

[8] Pooja Km, Samrat Mondal, and Joydeep Chandra. A graph combination with edge pruning-based approach for author name disambiguation. *Journal of the Association for Information Science and Technology*, 71(1):69–83, 2020.

[9] KM Pooja, Samrat Mondal, and Joydeep Chandra. Exploiting similarities across multiple dimensions for author name disambiguation. *Scientometrics*, 126(9): 7525–7560, 2021.

[10] Li Tang and John Walsh. Bibliometric fingerprints: name disambiguation based on approximate structure equivalence of cognitive maps. *Scientometrics*, 84(3): 763–784, 2010.

[11] Dongwook Shin, Taehwan Kim, Joongmin Choi, and Jungsun Kim. Author name disambiguation using a graph model with node splitting and merging based on bibliographic information. *Scientometrics*, 100(1):15–50, 2014.

[12] Arvin Varadharajalu, Wei Liu, and Wilson Wong. Author name disambiguation for ranking and clustering pubmed data using netclus. In *Australasian joint conference on artificial intelligence*, pages 152–161. Springer, 2011.

[13] Stephen B Johnson, Michael E Bales, Daniel Dine, Suzanne Bakken, Paul J Albert, and Chunhua Weng. Automatic generation of investigator bibliographies for institutional research networking systems. *Journal of biomedical informatics*, 51:8–14, 2014.

[14] KM Pooja, Samrat Mondal, and Joydeep Chandra. An unsupervised heuristic based approach for author name disambiguation. In *2018 10th International Conference on Communication Systems & Networks (COMSNETS)*, pages 540–542. IEEE, 2018.

[15] Humaira Waqas and Muhammad Abdul Qadir. Multilayer heuristics based clustering framework (mhcf) for author name disambiguation. *Scientometrics*, 126 (9):7637–7678, 2021.

[16] Emiel Caron and Nees Jan van Eck. Large scale author name disambiguation using rule-based scoring and clustering. In *Proceedings of the 19th international conference on science and technology indicators*, pages 79–86. CWTS-Leiden University, Leiden, 2014.

[17] Debarshi Kumar Sanyal, Plaban Kumar Bhowmick, and Partha Pratim Das. A review of author name disambiguation techniques for the pubmed bibliographic database. *Journal of Information Science*, 47(2):227–254, 2021.

[18] Vetle I Torvik and Neil R Smalheiser. Author name disambiguation in medline. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(3):1–29, 2009.

[19] Wanli Liu, Rezarta Islamaj Doğan, Sun Kim, Donald C. Comeau, Won Kim, Lana Yeganova, Zhiyong Lu, and W. John Wilbur. Author name disambiguation for pubmed. *Journal of the Association for Information Science and Technology*, 65(4): 765–781, 2014. doi: https://doi.org/10.1002/asi.23063. URL `https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.23063`.

[20] Kunho Kim, Athar Sefid, Bruce A Weinberg, and C Lee Giles. A web service for author name disambiguation in scholarly databases. In *2018 IEEE International Conference on Web Services (ICWS)*, pages 265–273. IEEE, 2018.

[21] Donghong Han, Siqi Liu, Yachao Hu, Bin Wang, and Yongjiao Sun. Elm-based name disambiguation in bibliography. *World Wide Web*, 18(2):253–263, 2015.

[22] Jian Wang, Kaspars Berzins, Diana Hicks, Julia Melkers, Fang Xiao, and Diogo Pinheiro. A boosted-trees method for name disambiguation. *Scientometrics*, 93(2): 391–411, 2012.

[23] Andreas Rehs. A supervised machine learning approach to author disambiguation in the web of science. *Journal of Informetrics*, 15(3):101166, 2021.

[24] Zaqqi YAMANI, Siti NURMAINI, Winda Kurnia SARI, et al. Author matching using string similarities and deep neural networks. In *Sriwijaya International Conference on Information Technology and Its Applications (SICONIAN 2019)*, pages 474–479. Atlantis Press, 2020.

[25] Hung Nghiep Tran, Tin Huynh, and Tien Do. Author name disambiguation by using deep neural network. In *Asian conference on intelligent information and database systems*, pages 123–132. Springer, 2014.

[26] Mark-Christoph Müller. Semantic author name disambiguation with word embeddings. In *International conference on theory and practice of Digital Libraries*, pages 300–311. Springer, 2017.

[27] Jie Tang, Alvis CM Fong, Bo Wang, and Jing Zhang. A unified probabilistic framework for name disambiguation in digital library. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):975–987, 2011.

[28] Hao Wu, Bo Li, Yijian Pei, and Jun He. Unsupervised author disambiguation using dempster–shafer theory. *Scientometrics*, 101(3):1955–1972, 2014.

[29] Ziyue Qiao, Yi Du, Yanjie Fu, Pengfei Wang, and Yuanchun Zhou. Unsupervised author disambiguation using heterogeneous graph convolutional network embedding. In *2019 IEEE international conference on big data (Big Data)*, pages 910–919. IEEE, 2019.

[30] Michael Levin, Stefan Krawczyk, Steven Bethard, and Dan Jurafsky. Citation-based bootstrapping for large-scale author disambiguation. *Journal of the American Society for Information Science and Technology*, 63(5):1030–1047, 2012.

[31] Yumin Zhu and Qingzhong Li. Enhancing object distinction utilizing probabilistic topic model. In *2013 International Conference on Cloud Computing and Big Data*, pages 177–182. IEEE, 2013.

[32] Jianyu Zhao, Peng Wang, and Kai Huang. A semi-supervised approach for author disambiguation in kdd cup 2013. In *Proceedings of the 2013 KDD Cup 2013 Workshop*, KDD Cup '13, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450324953. doi: 10.1145/2517288.2517298. URL `https://doi.org/10.1145/2517288.2517298`.

[33] Neil R Smalheiser, Vetle I Torvik, et al. Author name disambiguation. *Annual review of information science and technology*, 43(1):1, 2009.

[34] Ijaz Hussain and Sohail Asghar. A survey of author name disambiguation techniques: 2010–2016. *The Knowledge Engineering Review*, 32, 2017.

[35] Yutao Zhang, Fanjin Zhang, Peiran Yao, and Jie Tang. Name disambiguation in aminer: Clustering, maintenance, and human in the loop. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1002–1011, 2018.

[36] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):1–42, 2018.

[37] Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of NAACL-HLT*, pages 3543–3556, 2019.

[38] Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, 2019.

[39] Zhaoyang Niu, Guoqiang Zhong, and Hui Yu. A review on the attention mechanism of deep learning. *Neurocomputing*, 452:48–62, 2021.

[40] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1409.0473`.

[41] Yequan Wang, Minlie Huang, Xiaoyan Zhu, and Li Zhao. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 606–615, 2016.

[42] Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu. Attention-based transactional context embedding for next-item recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[43] Ronald A Rensink. The dynamic representation of scenes. *Visual cognition*, 7(1-3): 17–42, 2000.

[44] Maurizio Corbetta and Gordon L Shulman. Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience*, 3(3):201–215, 2002.

[45] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[47] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

[48] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057. PMLR, 2015.

[49] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[50] Peng Xu, Xiatian Zhu, and David A Clifton. Multimodal learning with transformers: A survey. *arXiv preprint arXiv:2206.06488*, 2022.

[51] Daniel W Otter, Julian R Medina, and Jugal K Kalita. A survey of the usages of deep learning for natural language processing. *IEEE transactions on neural networks and learning systems*, 32(2):604–624, 2020.

[52] Xiao Luo, Haoran Ding, Matthew Tang, Priyanka Gandhi, Zhan Zhang, and Zhe He. Attention mechanism with bert for content annotation and categorization of

pregnancy-related questions on a community q&a site. In *2020 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, pages 1077–1081. IEEE, 2020.

[53] Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. Pre-trained models: Past, present and future. *AI Open*, 2:225–250, 2021.

[54] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897, 2020.

[55] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. URL `http://arxiv.org/abs/1301.3781`.

[56] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[57] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[58] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. *Advances in neural information processing systems*, 28, 2015.

[59] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL `https://aclanthology.org/N18-1202`.

[60] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[61] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the*

2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics, 2019. doi: 10.18653/v1/n19-1423. URL `https://doi.org/10.18653/v1/n19-1423`.

[62] Ilya Makarov, Dmitrii Kiselev, Nikita Nikitinsky, and Lovro Subelj. Survey on graph embeddings and their applications to machine learning problems on graphs. *PeerJ Computer Science*, 7:e357, 2021.

[63] Palash Goyal and Emilio Ferrara. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78–94, 2018.

[64] Amr Ahmed, Nino Shervashidze, Shravan Narayanamurthy, Vanja Josifovski, and Alexander J Smola. Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*, pages 37–48, 2013.

[65] Sam T Roweis and Lawrence K Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[66] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in neural information processing systems*, 14, 2001.

[67] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[68] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[69] Kathi Canese and Sarah Weis. Pubmed: the bibliographic database. *The NCBI handbook*, 2(1), 2013.

[70] Vetle I Torvik, Marc Weeber, Don R Swanson, and Neil R Smalheiser. A probabilistic similarity metric for medline records: A model for author name disambiguation. *Journal of the American Society for information science and technology*, 56(2):140–158, 2005.

[71] Karen Sparck Jones. Index term weighting. *Information storage and retrieval*, 9(11): 619–633, 1973.

[72] Qian Zhou, Wei Chen, Weiqing Wang, Jiajie Xu, and Lei Zhao. Multiple features driven author name disambiguation. In *2021 IEEE International Conference on Web Services (ICWS)*, pages 506–515. IEEE, 2021.

[73] Mark-Christoph Müller, Florian Reitz, and Nicolas Roy. Data sets for author name disambiguation: an empirical analysis and a new resource. *Scientometrics*, 111(3): 1467–1500, 2017.

[74] Hui Han, Wei Xu, Hongyuan Zha, and C Lee Giles. A hierarchical naive bayes mixture model for name disambiguation in author citations. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 1065–1069, 2005.

[75] Aron Culotta, Pallika Kanani, Robert Hall, Michael Wick, and Andrew McCallum. Author disambiguation using error-driven machine learning with a ranking loss function. In *Sixth International Workshop on Information Integration on the Web (IIWeb-07), Vancouver, Canada*, 2007.

[76] Xuezhi Wang, Jie Tang, Hong Cheng, and S Yu Philip. Adana: Active name disambiguation. In *2011 IEEE 11th international conference on data mining*, pages 794–803. IEEE, 2011.

[77] Yanan Qian, Qinghua Zheng, Tetsuya Sakai, Junting Ye, and Jun Liu. Dynamic author name disambiguation for growing digital libraries. *Information Retrieval Journal*, 18(5):379–412, 2015.

[78] Humaira Waqas and Abdul Qadir. Completing features for author name disambiguation (and): an empirical analysis. *Scientometrics*, pages 1–25, 2022.

[79] Li Zhang. LAGOS-AND: A Large, Gold Standard Dataset for Scholarly Author Name Disambiguation, February 2021. URL `https://doi.org/10.5281/zenodo.4568624`.

[80] Lada A Adamic and Bernardo A Huberman. Zipf's law and the internet. *Glottometrics*, 3(1):143–150, 2002.

[81] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4): 671–687, 2003.

[82] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[83] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[84] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[85] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[86] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL `http://arxiv.org/abs/1412.6980`.

[87] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017.

[88] Qingyun Sun, Hao Peng, Jianxin Li, Senzhang Wang, Xiangyu Dong, Liangxuan Zhao, S Yu Philip, and Lifang He. Pairwise learning for name disambiguation in large-scale heterogeneous academic networks. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 511–520. IEEE, 2020.

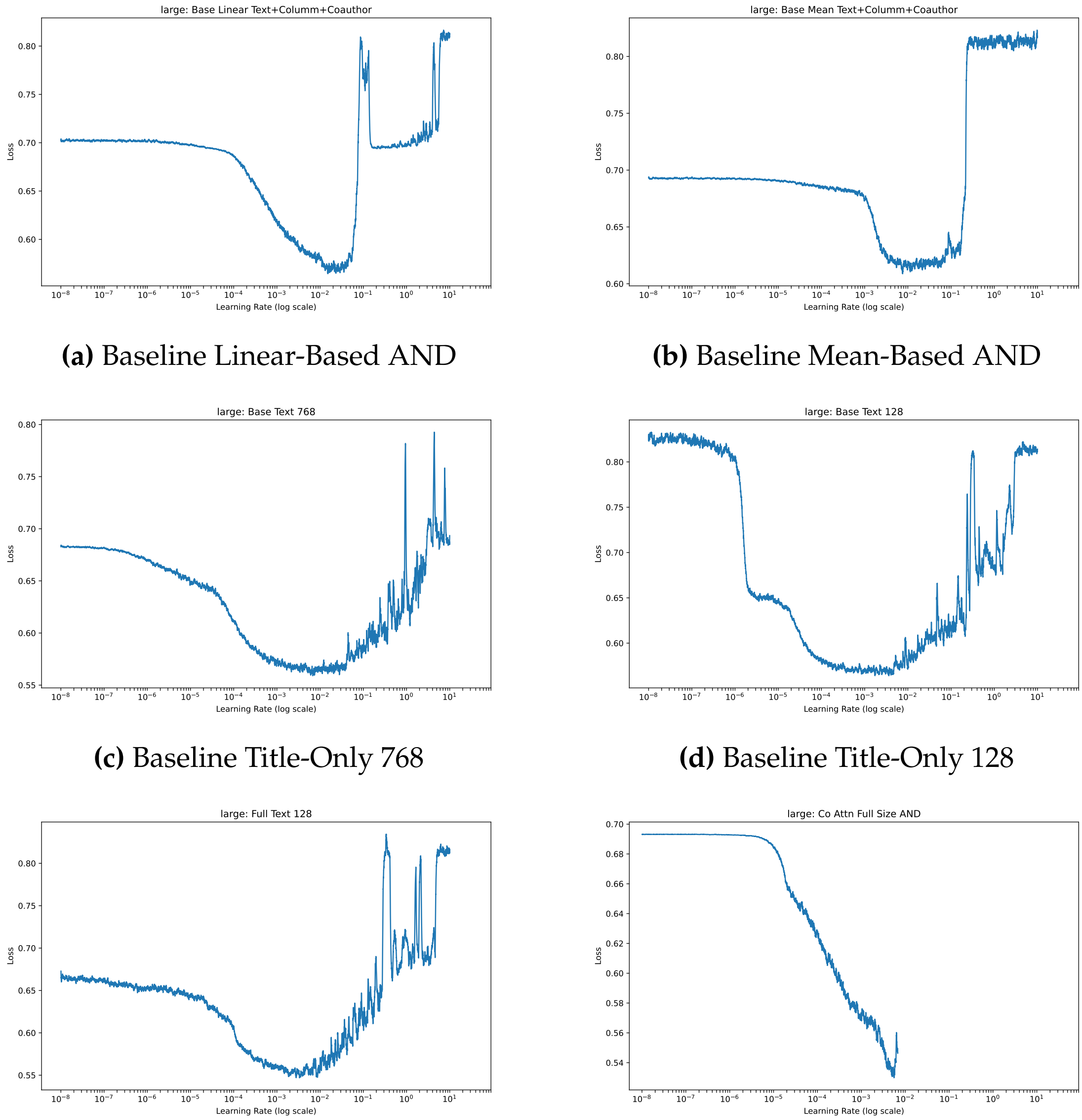
[89] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8): 861–874, 2006.

[90] Rob Koopman, Shenghui Wang, and Gwenn Englebienne. Fast and discriminative semantic embedding. *IWCS 2019*, page 235, 2019.

A Example of A Record

```
##TTLlevel 0
##TTLnumber 790
##TTLisactive 0
##TTLstatus 1
##TTLtype standard
$001@ $02-4,9,11,16,74,108,331
$001A $00001:06-07-78
$001B $00498:27-03-15$t01:00:49.000
$001D $09999:99-99-99
$001U $0utf8
$001X $00$A96$A97
$001Z $01023
$002@ $0Aax
$002C $atekst$btxt$2rdacontent/dut
$002D $azonder medium$bn$2rdamedia/dut
$002E $aband$bnc$2rdacarrier/dut
$003@ $0750007907
$003O $aOCoLC$0905671507
$004A $00416195407
$006B $0GB7522889
$010@ $aeng
$011@ $a1970
$013@ $0De
$019@ $agb
$021A $aThe @reader's encyclopedia of world drama$hed. by John Gassner & Edward Quinn
$028C/01 $dJohn$aGassner$9067855040
$028C/02 $dEdward$aQuinn$9067625622
$033A $pLondon$nMethuen
$034D $aXIII, 1030 p
$034M $aill
$037E $aOorspr. uitg.: New York : Crowell, 1969
$044A $S0$aDrama$vDictionaries
$044K/01 $9078691362
$044M/01 $9078001536
$044M/02 $9077973372
$044N/01 $9088144143
$045A $S##$aPN1625$b.R4 1970
$045F $S04$g18$a809.2
$045J/01 $a821
$045Q/01 $9077600029
```

B Learning Rates Finding Results on Record Splitting Datasets



(a) Baseline Linear-Based AND

(b) Baseline Mean-Based AND

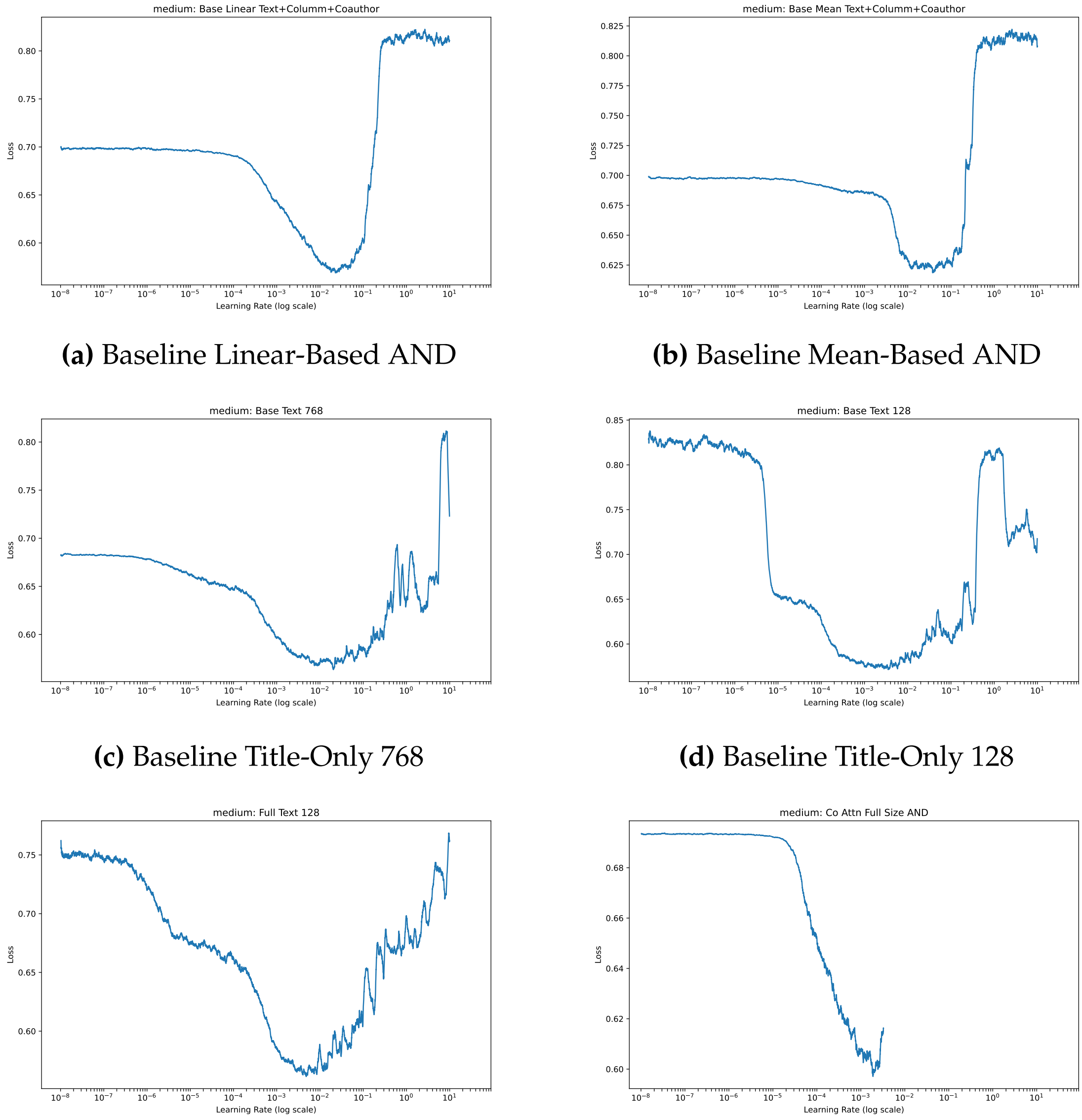
(c) Baseline Title-Only 768

(d) Baseline Title-Only 128

(e) Baseline Full-Text 128

(f) Co-Attention-Based AND

Figure B.1: Learning rates of all models on the large dataset using the record-splitting method.



(a) Baseline Linear-Based AND

(b) Baseline Mean-Based AND

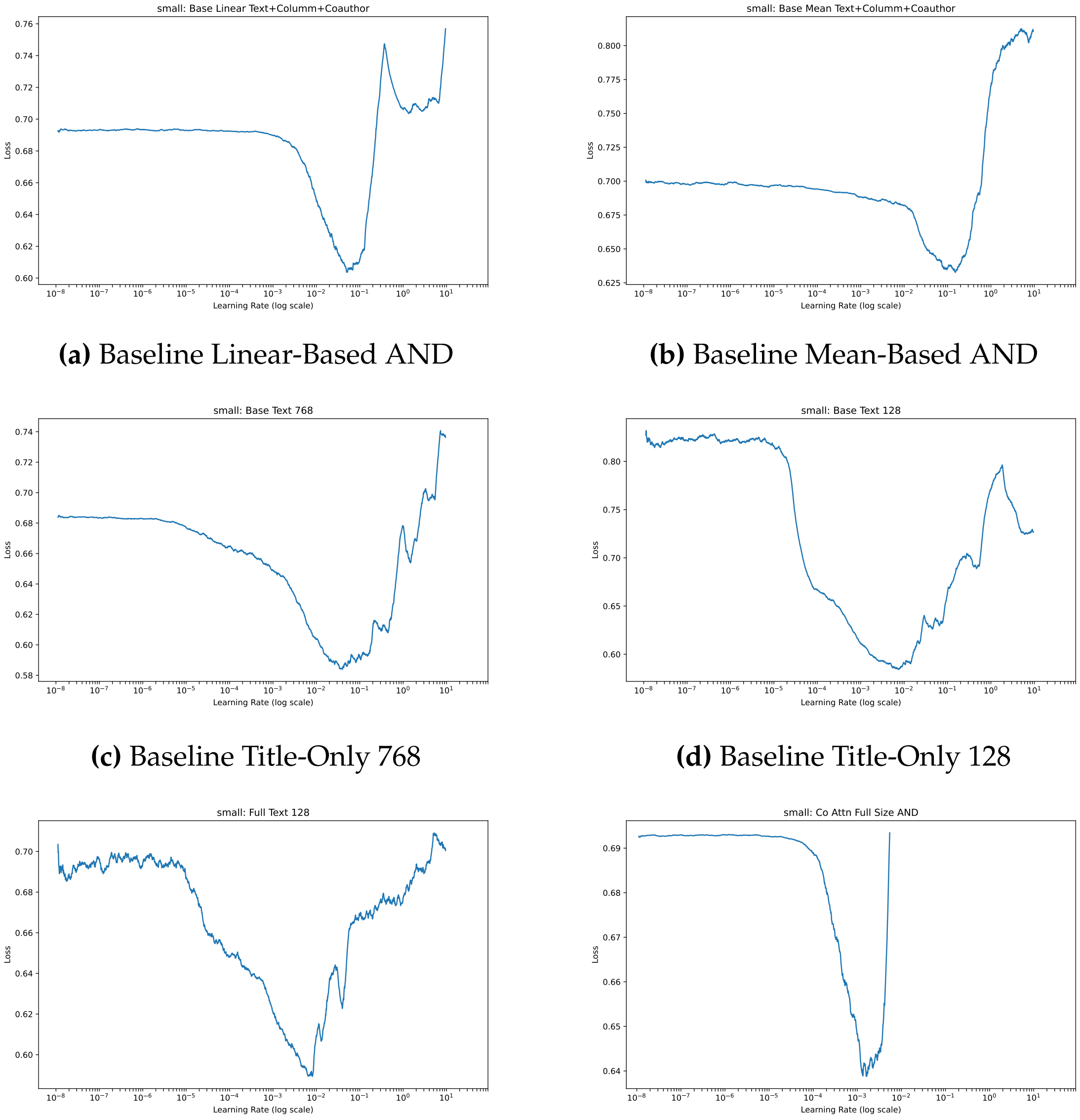
(c) Baseline Title-Only 768

(d) Baseline Title-Only 128

(e) Baseline Full-Text 128

(f) Co-Attention-Based AND

Figure B.2: Learning rates of all models on the medium dataset using the record-splitting method.



(a) Baseline Linear-Based AND

(b) Baseline Mean-Based AND

(c) Baseline Title-Only 768

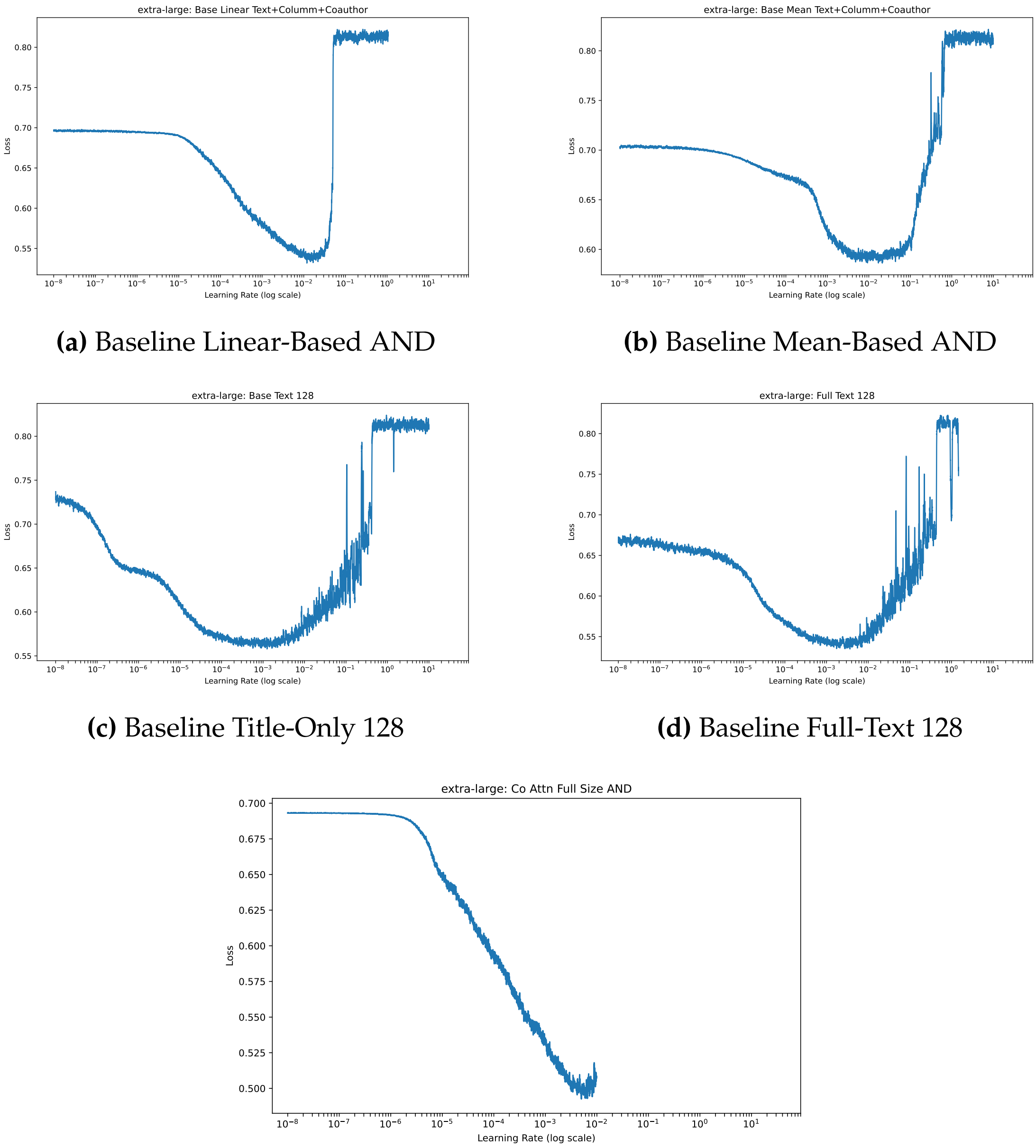
(d) Baseline Title-Only 128

(e) Baseline Full-Text 128

(f) Co-Attention-Based AND

Figure B.3: Learning rates of all models on the small dataset using the record-splitting method.

C Learning Rates Finding Results on Block Splitting Datasets



(a) Baseline Linear-Based AND

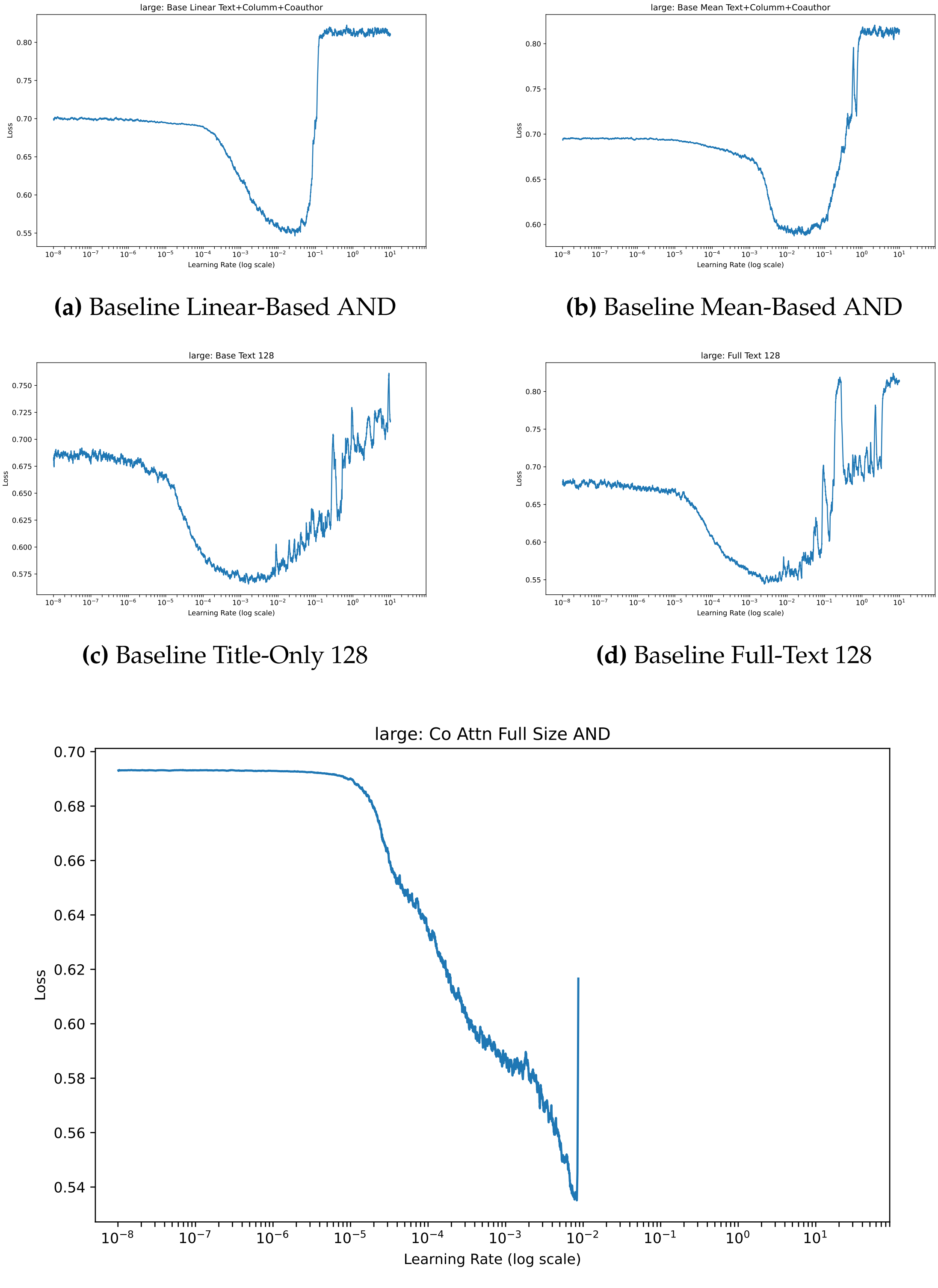
(b) Baseline Mean-Based AND

(c) Baseline Title-Only 128

(d) Baseline Full-Text 128

(e) Co-Attention-Based AND

Figure C.1: Learning rates of all models on the extra-large dataset using the block-splitting method.



(a) Baseline Linear-Based AND

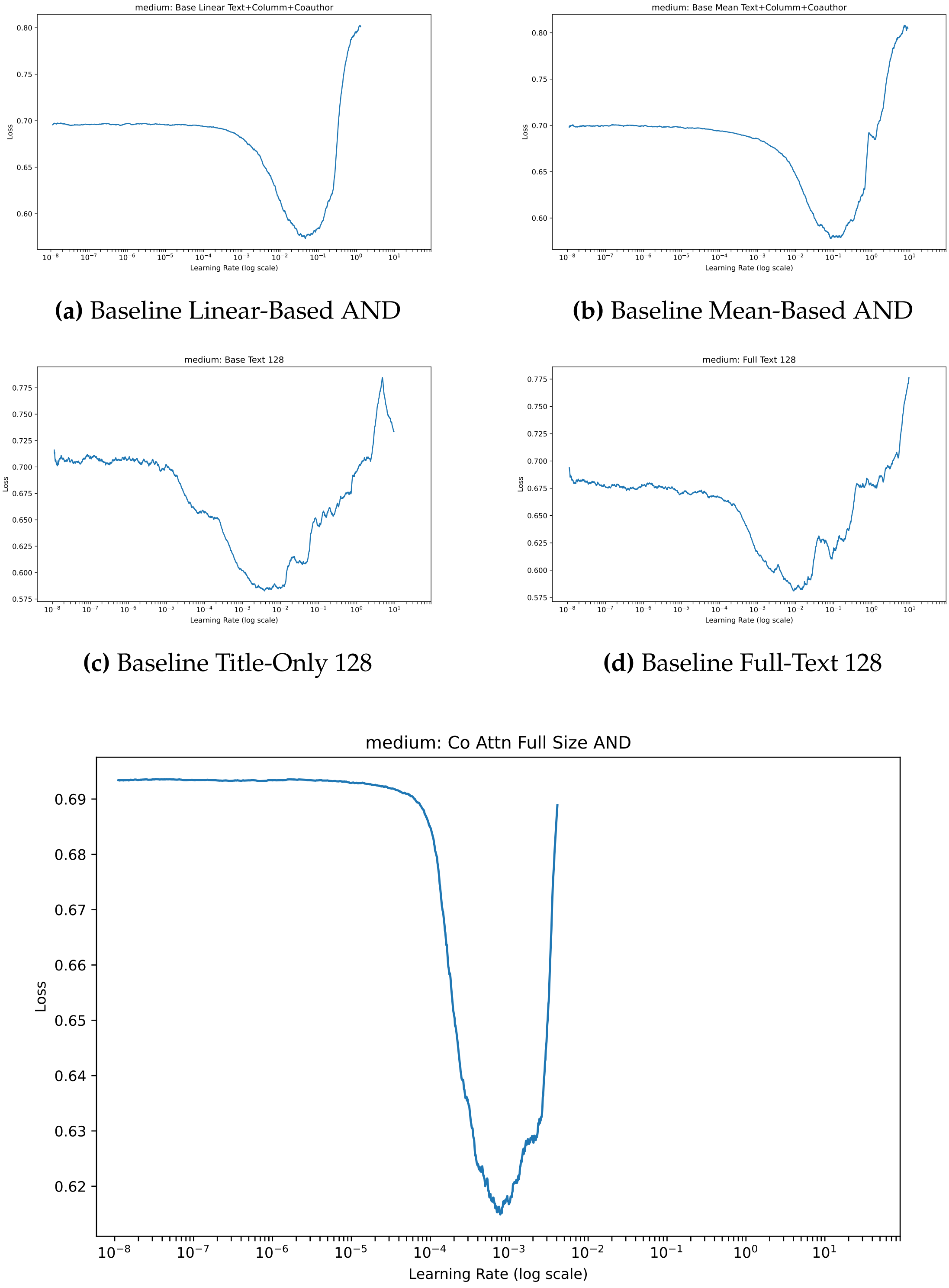
(b) Baseline Mean-Based AND

(c) Baseline Title-Only 128

(d) Baseline Full-Text 128

(e) Co-Attention-Based AND

Figure C.2: Learning rates of all models on the large dataset using the block-splitting method.



(a) Baseline Linear-Based AND

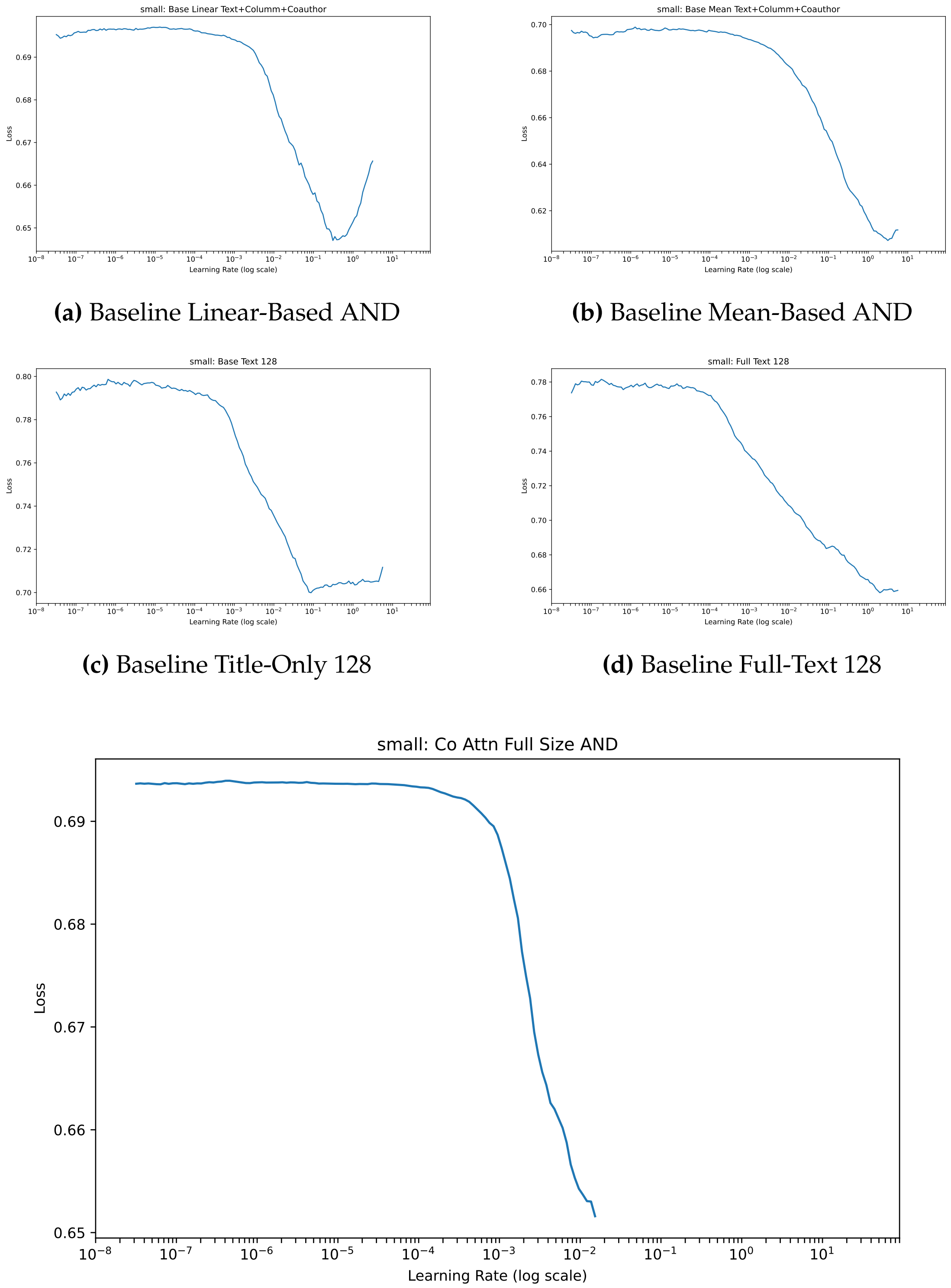
(b) Baseline Mean-Based AND

(c) Baseline Title-Only 128

(d) Baseline Full-Text 128

(e) Co-Attention-Based AND

Figure C.3: Learning rates of all models on the medium dataset using the block-splitting method.



(a) Baseline Linear-Based AND

(b) Baseline Mean-Based AND

(c) Baseline Title-Only 128

(d) Baseline Full-Text 128

(e) Co-Attention-Based AND

Figure C.4: Learning rates of all models on the small dataset using the block-splitting method.

D Training Time of All Models on Block splitting Datasets

Table D.1: The models' training time of one epoch on the block-splitting datasets.

Model	Dataset			
	Small	Medium	Large	Extra-Large
Co-Attn-Based AND	6s	52s	8m35s	49m27s
Linear-Based AND	2s	29s	4m52s	31m04s
Mean-Based AND	3s	26s	4m46s	30m12s
Title-Only AND	2s	19s	2m48s	14m51s
Full-Text AND	1s	18s	2m55s	15m36s

E Confusion Matrices on The Extra-large Dataset Using The Block-splitting method

Table E.1: The confusion matrix of the co-attention-based AND model on the extra-large dataset using the block-splitting method.

		Actual Class	
		Positive (P)	Negative (N)
Predicted Class	Positive (P)	234,441	20,793
	Negative (N)	25,917	239,565

Table E.2: The confusion matrix of the linear-based AND model on the extra-large dataset using the block-splitting method.

		Actual Class	
		Positive (P)	Negative (N)
Predicted Class	Positive (P)	211,993	20,661
	Negative (N)	48,365	239,697

Table E.3: The confusion matrix of the mean-based AND model on the extra-large dataset using the block-splitting method.

		Actual Class	
		Positive (P)	Negative (N)
Predicted Class	Positive (P)	174,937	62,292
	Negative (N)	85,421	198,066

Table E.4: The confusion matrix of the title-only AND model on the extra-large dataset using the block-splitting method.

		Actual Class	
		Positive (P)	Negative (N)
Predicted Class	Positive (P)	186,792	69,359
	Negative (N)	73,566	190,999

Table E.5: The confusion matrix of the full-text AND model on the extra-large dataset using the block-splitting method.

		Actual Class	
		Positive (P)	Negative (N)
Predicted Class	Positive (P)	188,059	51,589
	Negative (N)	72,299	208,769