# Sci-Bert: Semantic shift tracing in Scientific Literature by Clustering in BERT-based Embedding Space

Hamza Waheed

A report submitted as part of the requirements
for the degree of M.Sc in Big Data Analytics
at the School of Computing and Digital Technology
Birmingham City University, Birmingham, UK

March 2024

Supervisor Dr. Amina Dridi

# Abstract

Scientific literature has gone through linguistic evolution over the years. Investigating the evolution of scientific linguistic and understanding future trends can lead to better resource allocation. Organizations, academic institutions and research practitioners can analyse these trends to make informed decisions on grant, fund allocation and selection of research topics for the future. A lot of work has been produced to analyse the temporal evolution in academia by investigating the dynamism of the keywords over time. Most research involves Bibliometric and Natural Language Processing (NLP) techniques. However, these studies doesn't account for the semantic context of words under consideration. Unlike using word embedding techniques like word2vec, glove, that generate single vector representation of a word with different senses. This report proposes Sci-Bert, that uses BERT, a transformer based word embedding approach, to investigate the changing dynamism of keywords with contextual awareness. The main contribution of this report is to examine the contextual dependence/independence of keywords in scientific literature. The embeddings are generated in such a way that, each sentence (context), a word appears in, is taken into consideration. This helps to generate contextual embeddings, in a manner that preserves the semantic information of the keyword of interest. The generated embeddings are modelled using K-nearest neighbour (K-NN) to explore the changing neighbourhoods. The changing neighbourhood of words over time helps to identify the dynamism of the domain of interest (keyword) in a temporal fashion. For the proposed research, scientific corpora is generated using the papers published in the Neural Information Processing Systems (NIPS; actually abbreviated NeurIPS) conference between 1987 and 2016 in the domain of Machine Learning (ML). A comparative analysis is drawn with the paper "Vec2Dynamics: A Temporal Word Embedding Approach to Exploring the Dynamics of Scientific Keywords" to examine contextual dependence of keywords in scientific literature. Sci-Bert algorithm produces remarkable results in tracing the dynamism of keywords in temporal fashion. The results obtained suggest a general strong consistency with the timeline of machine learning, implying the high efficacy of BERT in tracing the shift in the domain of ML. The findings of the report highlight BERT being able to effectively identify and map the techniques (algorithms) to the studied domain and stays consistent with the results obtained by Vec2dynamics.

# Acknowledgements

I would like to express my gratitude to my supervisor Dr. Amna Dridi for her continuous support and weekly feedbacks. This project would not have been possible without her help and effort. I would also like to thank Dr Atif Azad and Dr Amna Dridi for supporting me during the selection phase of my project. Lastly, I wish to thank my family for being supportive of me.

# Contents

# List of Tables

# List of Figures

# Chapter 1

# Introduction

## 1.1 Background And Rationale

Human language is an evolving construct, where the word semantics and their association evolve over time. Change in word semantics is reflected in the usage of the word. As more people use the word in different context, the meaning of the word changes over time. As an example, the word "Apple" was associated with fruits, but in contemporary linguistics "Apple" is also associated with business and technology Yao et al. (2018a).

As a result of constant evolution, linguistic studies have become an active research area for scientists and researchers. The linguistic evolution helps better understand and unravel the cultural advancement in human history. The rules that cause the linguistic evolution are still unknown and are under study. But generally, advancement in languages often starts with the change in word semantics.

Study of Linguistic advancement has been a real challenge due to the traditional analytical approaches and the scarce availability of the data. The advancement in computational analysis approaches in the area of Computer Science and Machine Learning has made it possible to mine large amount of data for knowledge discovery tasks.

Similarly, linguistic evolution in scientific literature has also seen shift and is an active area of research for its practical implications in the scientific, financial and technological implications.

In recent years, semantic shift tracing in academia, as an area for computational analysis has become an active research domain, mainly for two reasons, 1) Availability of publicly accessible scholarly databases and digital catalogues on different search spaces. 2) For automation and better resource management by understanding the future evolution of research areas.

The importance of semantic shift tracing in academia is compounded by the need to make better informed decisions in funds allocation and resource allocation on emerging topics and trends that may yield better commercial applications.

To investigate and contribute towards the previously described reasons, many empirical studies target keyword count as a metric to map the emerging trends over time Pesta, Fuerst and Kirkegaard (2018). Despite showing promise, keyword count based approaches doesn't capture the essence of the content and are limited in scope.

Other works Han (2020a), Malaterre et al. (2021) use natural language processing techniques like Topic Modelling for investigating the evolution and impact of research topics. Topic modelling, as the name suggests, aims to extract topics by mapping the semantics of the document sentences and then extracting & modelling topics to the document. However, topic modelling techniques capture the semantics at document level and doesn't account for the underlying association between the words Dridi et al. (2022).

In recent years, Mathematical representation of languages at word level has been developed to preserve the underlying semantics of the word to better model natural language tasks that are understandable to the computer. Approaches like word embeddings have been developed to tackle the association between words Mikolov et al. (2013). These models capture the essence at word level and capture the similarity semantics of different words.

Mathematically, embedding is the process of squashing multidimensional vector space into low dimensions. In Natural language Processing (NLP), word embedding techniques explore the semantic relation at word level by projecting words in vector space. These vector representations are projections such that semantically similar words are geometrically closer in vector space. Unlike topic modelling that considers semantics at document level, word embedding techniques take into account the association at word level,

Many empirical studies Yao et al. (2018b), Hamilton, Leskovec and Jurafsky (2016) focus on the dynamics of keywords over time to detect semantic shift. These approaches take the context of the keywords, in particular, these studies take a look at n words before and after the keyword and apply word embedding techniques to generate vectors. However, when generating word embeddings using techniques such as glove, word2vec. The embedding vectors, generated for the keyword that has different word senses, would have single representation in vector space.

Studies conducted by Devlin et al. (2018), Beltagy, Lo and Cohan (2019), suggest that unsupervised pretraining language models such as BERT, ELMO display significant performance improvement over previously mentioned traditional approaches. BERT & ELMO use transformer architecture where an attention mechanism aids in calculating the probability of a word appearing given a sequence of words Vaswani et al. (2017). These models return contextualized word embeddings that can be utilized for machine learning and NLP tasks. They do so by generating different embedding vectors of the same keyword for different contexts. One advantage of these models is, these are pretrained, meaning the models have already been trained on large corpora, however, these models can be tailored to the project needs by fine-tuning on project specific corpora.

In this paper, we employ "Sci-Bert" algorithm, a novel approach that uses context aware word embedding technique, BERT, to check the semantic shift of keywords in academic literature over time, mainly in the domain of "Machine Learning". This novel approach, "Sci-Bert", is a neural network (Transformer) based approach that uses, BERT based contextualized temporal word embedding.

Sci-Bert takes into account lexical context of the word in the scientific corpora. Contextualized word embeddings are generated using BERT for each word in the vocabulary for each word appearing in each time slice. Then, we define K-nearest neighbours of these generated embeddings based on similarity measure. In the end, the evaluation of the model is carried out by stability measure. A detailed look into the algorithm is given in Methodology chapter 3.

This paper is an extension of the work "Vec2Dynamics: A Temporal Word Embedding Approach to Exploring the Dynamics of Scientific Keywords—Machine Learning as a Case Study" Dridi et al. (2022). The experimental methodology followed in this paper is similar to the approach applied in Dridi et al. (2022). The rationale following similar methodology is for the result comparison between "Sci-Bert" and "Vec2Dynamics". This would help in investigating the impact of both BERT and word2vec based embedding on changing dynamics in scientific literature.

This paper provides the following contribution to the community:

1. Generates BERT based contextualized word embedding over scientific corpora, mainly in the domain of Machine Learning.
2. Generates contextualized embeddings of each word by taking the context (sentence) into account.
3. Applies similarity measures to explore the similarity of generated word embeddings (vectors).
4. Perform K-NN to investigate the changing neighbourhood of keywords over time.
5. Investigate and discuss the effect of contextual word embedding on dynamics of the keywords in the scientific literature.
6. Compares the results of Cosine Similarity and Dot Product Similarity.
7. Provides a visual and intuitive guide on the changing dynamism of the keywords.

Lastly, the potential applications and the impact of the paper include but not limited to:

1. Improve categorization of research catalogues in digital libraries based on the keywords.

2. Medical and Finance domain, where the technical terms are dependent on context.
3. Help in automating the documentation process in industrial and scientific domain.
4. Facilitation in making informed decisions on future emerging trends and fund allocation in academia.
5. Assessing quality of the work based on the semantic shift of keywords in different versions of the same paper.

## 1.2 Research Aim

The research aims to investigate the temporal dynamism in scientific literature especially in the domain of Machine Learning by employing transformer based architecture namely, BERT for generating contextual word embedding.

## 1.3 Research Objectives

To accomplish the research aim of this study, the following objectives are derived.

1. Generate BERT based contextual word embedding of keywords, to get the vector representation over different time periods.
2. Investigate the role of similarity measure, namely cosine similarity, to determine the closeness of generated word vector embeddings.
3. Develop the K-NN model and examine the overlapping keywords in subsequent time windows.
4. Evaluate and analyse the stability of the K-NN algorithm over subsequent time windows.
5. Investigate the overlapping neighbours in subsequent time windows to explore the semantic shift.
6. Compare the results with the seminal paper Dridi et al. (2022).

## 1.4 Research Hypothesis

The research is carried out under the hypothesis, when word embeddings are generated using BERT, mainly when we take context (sentence) of each word into consideration, then due to the different word senses of a keyword being represented by different word vectors in BERT. This would cause variational stability of K-NN, with fewer overlaps of keywords within the subsequent time windows. The assumption is, instead of producing the same results as word2vec, Sci-Bert would represent increased dynamism of keywords over time.

# Chapter 2

# Technical Background and Related Work

This section provides a succinct technical background and reviews empirical work in capturing the latest trends in the area of semantic shift detection. Due to the context of the research, the report discusses the Transformer architecture that is the heart of the BERT model used in Sci-Bert algorithm.

## 2.1 Artificial Neural Network

Artificial Neural Networks (ANN) are the labelled directed graph structures that draw its inspiration from the biological neural networks. In recent years, ANN have gained massive adaptation and popularity because of state-of-the-art results in regression, clustering, classification and pattern recognition tasks.

ANN comprise neurons that are organized in interconnected layers Dongare, Kharde and Kachare (2012). Each node in the layer performs some calculation and passes its output to the next layer. The connection between the node conveys the signal that is either amplified or reduced depending on the weight attached to the signal.

ANN are composed of three layers: Input layer, Hidden Layer and Output Layer. Input unit gets the data from the external source. Hidden layers present between the input and output layer perform mathematical operations to conform the input data to minimize some loss function for the objective of the task at hand. Output layers perform the last adjustments to conform the output to the desired shaped output Wu and Feng (2018). Due to its high preference and efficient processing in parallel implementation systems, many variations of ANN have been developed to aid in scientific research, image & signal processing and natural language processing tasks Abiodun et al. (2018).

In early days of Machine Learning, ANN techniques were limited in scope to process raw, unstructured data that was present in different forms. In recent years, ANNs have seen a leap in development owing to the rigorous research and efforts of researchers. The progress resulted in devising representation learning algorithms that involve a machine to automatically discover patterns in the raw data LeCun, Bengio and Hinton (2015). This representational learning has developed into a subset of Machine Learning (ML) called Deep Learning (DL).

DL is a subset of ML, that uses ANN with complex multilayer structure. The distinction between traditional ANN and DL methods lie in more complex ways of connecting neurons and layers that can model complex information systems Abiodun et al. (2018).

Due to efficacy, modelling complex systems and robustness, many approaches in DL have been developed over the years to solve different problems. The following gives a list of popular approaches in DL;

1. Convolutional Neural Network (CNN)
2. Recurrent Neural Network (RNN)
3. Boltzmann Machines
4. Graph Networks
5. Transformers

As, the report uses Google BERT (Bidirectional Encoder Representation For Transformer) for semantic shift tracing in temporal fashion. The rest of the section will focus on Transformers, that is at the heart of Google BERT. However, some references to other approaches will be made to distinguish the key features of transformers compared to other approaches.

## 2.2 Transformers

Sequential Learning is the ability to encode and represent the sequence of discrete elements and occurrences for better interpretation. Sequential Learning plays a key role in human language processing and tasks with sequential actions Conway and Christiansen (2001).

Sequential learning has also been an integral part of DL. RNN, Long short-term memory (LSTM) Hochreiter and Schmidhuber (1997) have long been the state-of-the-art methods for sequential modelling tasks in machine learning such as Natural Language processing (NLP) and Time Series Analysis. These sequential approaches take input one at a time and produce the hidden states (output). The previous state output is the input of the next state. The model, then, remembers (manages order of sequence internally) the sequence of the inputs. This helps the model to decipher the parts of the sequence that are important.

The problem with these approaches arise with the sequential dependence on the input. This sequential nature impedes the parallelism when training the model, making the models slow and computationally expensive. The models also suffer from the vanishing gradient problem. For a multi-layered neural network with multiple activation functions, the back-propagated error by gradient descent, starts getting smaller when it approximately approaches 0 that makes fine-tuning the weights of the connections and training the model hard Hochreiter (1998).

In 2017, the research team at Google Research and Google Brian published the paper "Attention Is All You Need" Vaswani et al. (2017). The proposed method tackles the performance, parallelism and computational aspect of the previous approaches. The paper proposed a novel approach, "Transformer", that rely on self-attention mechanism. The proposed approach, instead of taking input sequentially, takes all the input vectors (embeddings) at once along with their positional embeddings that encode the relative position of the words in the sentence. The next section briefly discusses the architecture of the transformer.

### 2.2.1 Architecture

Sequential Neural models are often based on encoder-decoder architecture. The input sequence is mapped to another representation by the encoder layer. The decoder layer then generates the output sequence from the produced input representation. The model is autoregressive, meaning that the model uses the previously generated output as an input sequence for the next step. The model follows the encoder-decoder architecture that comprises, stack of multi-head attention layers in both encoder and decoder parts. The architecture of the transformer is shown in fig 2.1.

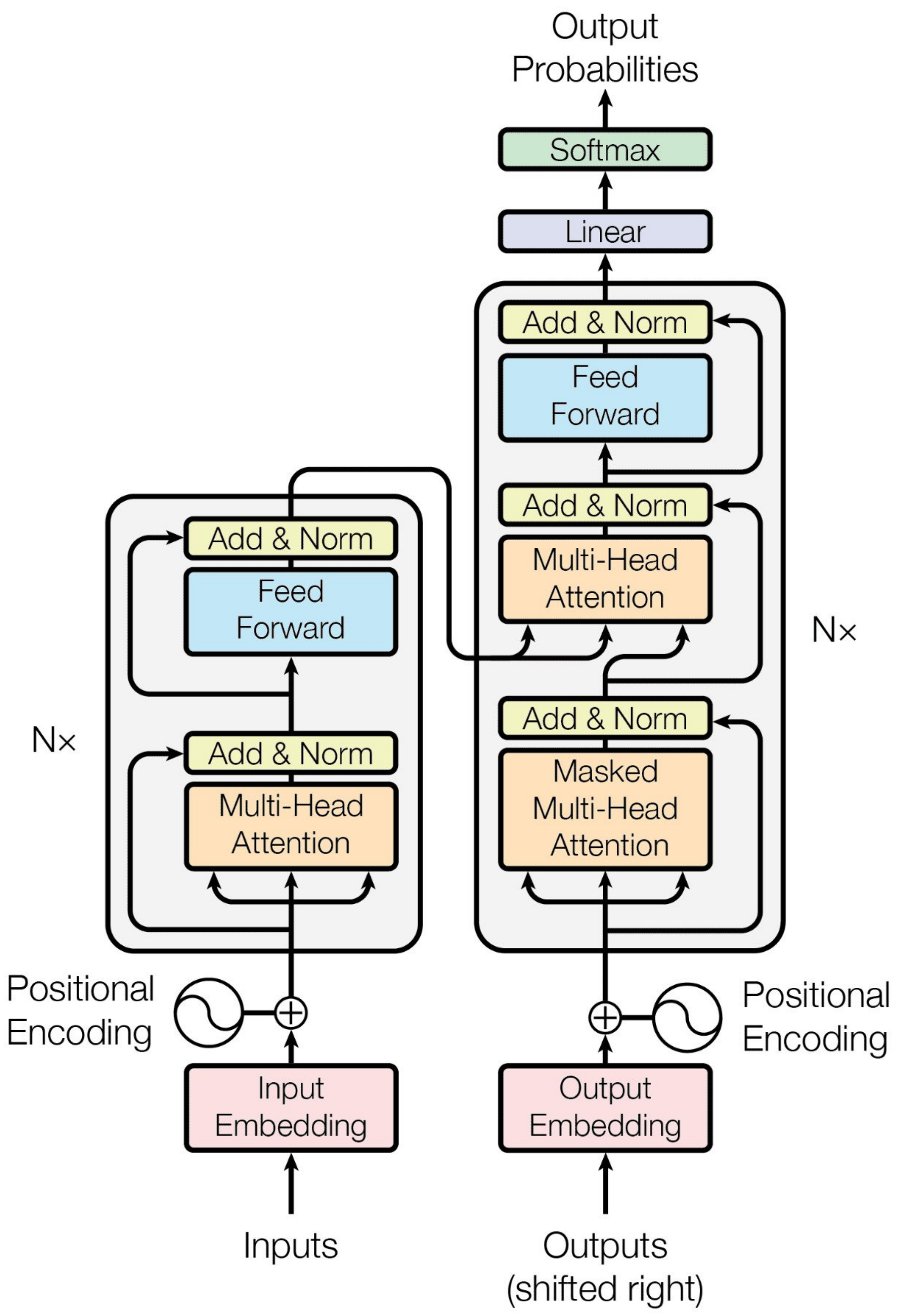


Figure 2.1: Transformer Architecture Vaswani et al. (2017)

Unlike RNN and LSTM, transformer doesn't take input as a sequence but as a whole. This give rise to two main problems: First, the order the words appear in is lost. This can lead to significant interpretation issues of the sentences. For example, "George did not win the match, but he was satisfied". Now take a look at the same sentence with slightly different word positions. "George did win the match, but he was not satisfied". Both sentences convey totally different meaning, therefore it's imperative to keep track of the word order.

The second problem is to deal with preserving the context of the sentence. The problem is to make the model context aware. For example, "the Robber robbed the bank, then he went to the bank of the river and sat idly". The problem is to make the model differentiate between the two "bank" that serve completely different meanings depending on the context. The issue is to make the model aware which words to pay most attention to in context to a given word.

To resolve the problems, the authors of the seminal paper "Attention is all you need" employed two approaches. The first technique was to use positional embeddings for each word to store the positional information alongside the vector embedding of the word. Second, the authors introduced a multi-head attention layer called "Self-Attention". Multi-head attention layer consists of many "single Scaled Dot-Product attention layer". Each single attention layer input consists of queries, keys and values. The dot product between the queries (Q) and keys (K) is computed, and the softmax function is applied, the resulting scores obtained from the operation are compared to the values (V) i.e. the word embeddings. that best match the scores. This way, the words that have high similarity score with each other are given the most attention. The same process is repeated across multiple layers, hence the multi-head attention layer. The whole architecture of the attention layer is illustrated in the figure 2.2.

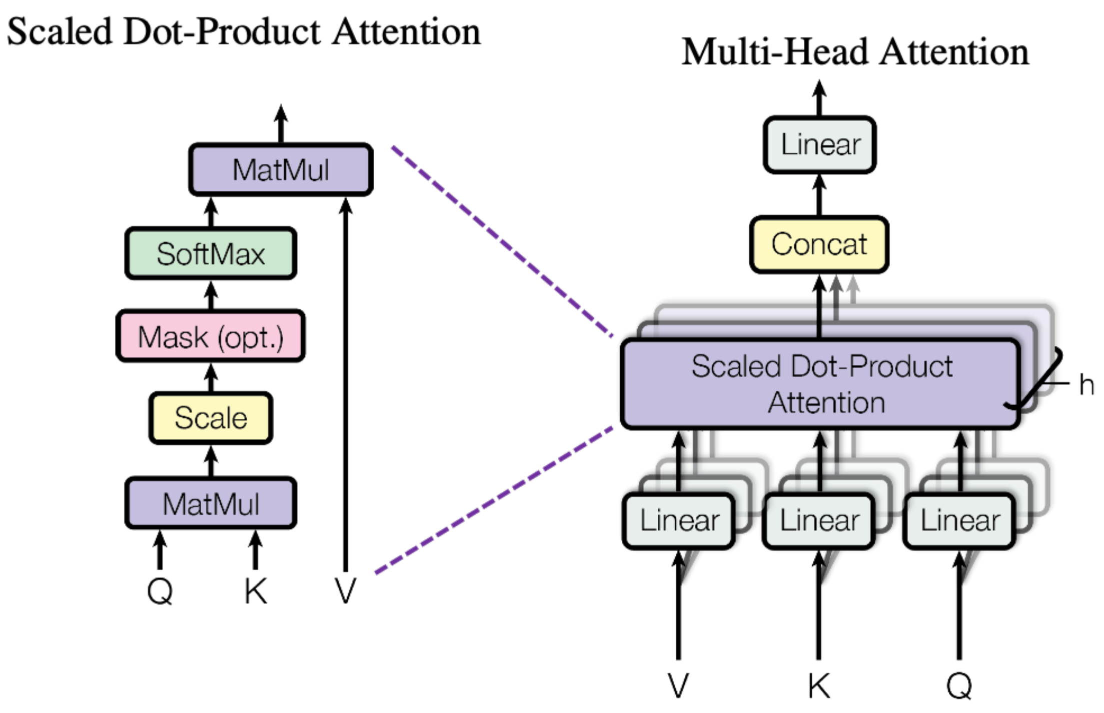


Figure 2.2: Multi-head attention layer in Transformer Vaswani et al. (2017)

Just like encoder, decoder follows the same design. The only difference in encoder and decoder is, the input of the multi=head attention in the decoder is the generated output of the encoder stacks. Each decoder layer is followed by the normalization layer. Also, in each sublayer of multi-head attention in the decoder, some of the positions are masked.

The Transformer achieved state-of-the-art results in NLP domain, mainly in the area of language translation that require semantic and context awareness for translation. Unprecedented success of Transformers have made it an established architecture for language models, The Transformer is at the heart of Google BERT.

## 2.3 BERT

In 2018, the research team at Google introduced a new language representation model, BERT (Bi-directional Encoder Representations from Transformers). Unlike other language models, BERT is a pre-trained model, but it could be fine-tuned for language specific tasks. The advantage of using BERT is, it's a pretrained model and requires very minimal configuration to get started. Additionally, BERT was trained on unlabelled text for the task of "question answering" and "masked language modelling", aiding the contextual awareness of the embeddings Devlin et al. (2018).

Since inception BERT has been trained on variety of tasks such as text classification, masked language modelling and language translation, text summarization and text prediction based on context in Gmail Martinc, Kralj Novak and Pollak (2020).

In this report, we use BERT for its state-of-the-art performance in capturing semantic context in language aspects Reddy, Singh and Srivastava (2020). The report proposes SciBert a novel algorithm, as an alternative to the traditional Bibliometric, non-contextualized embedding techniques to analyse the semantic shift of words in temporal fashion. The next section elaborates on the previous research techniques used by researchers to investigate the diachronic shift in literature. We go over the relevant papers and discuss the research gap this paper tries to fill.

## 2.4 Related Work

## 2.5 Bibliometric & Content Based Approaches

Fiala and Tutoky (2017) presented bibliometric analysis approach to analyse the trends involved in computer science domain. They used 1.9 million papers published from 1945 to 2014 indexed on Web of Science. The experiment involved performing exploratory and statistical analysis by running queries against the dataset to map the popular topics during different time windows. Their findings identified "Ontology", "Genetic Algorithms", "Cloud Computing" being the popular trends in empirical literature.

Marín-Marín et al. (2019) used bibliometric analysis on the scientific literature produced in academia on "Big Data". The study comprised of papers from digital libraries: Web of Science (WOS), Scopus, ERIC, and PsycINFO that included the term "Big Data" in the title, abstract, keyword and content. The paper applied citation count and graph analysis to examine the diachronic change. The paper applied network graph analysis to compute raw frequency of words to trace the semantic shift and relationship of keywords. The graph analysis yielded three interconnected clusters representing the approaches, domain and institution of the paper. The experiment showed that "Machine Learning", "Big Data Analytics", "Cloud Computing" were some hot key terms in the graph analysis.

Han (2020b) studied evolution of research topics by employing text based approach on Library and Information System (LIS). The study applied Latent Dirichlet allocation (LDA), a topic modelling technique, to identify the subject areas of the research. The proposed approach investigated the evolution and impact of research topics. The results reveal "Information Retrieval" and "Information Science" being the dominant topics over 5 time windows. The study suggests that the topics that appeared once in the analysis had undergone context change due to the evolving field of computer science over earlier time periods.

Popescu and Strapparava (2015) employed four techniques based on a novel Diachronic Text Evaluation approach, that include identifying the time period of a keyword and google n-gram to check pairwise association, named-entities search. The approach used Support vector machine (SVM) to classify a word into multi-class time window when the keyword undergoes semantic change. The result shows promise, but include error bars when predicting time windows of the word.

The above discussed approaches show merit by performing statistical, bibliometric and content-based approaches. All these approaches fail to encapsulate the linguistic context the word appears in.

## 2.6 Word Embedding & Distributional Models

Distributional methods using embedding vectors to represent the semantic shift of words has shown massive improvement over bibliometric, statistical and word frequency count methods.

Frermann and Lapata (2016) capture the evolving word sense over time by introducing dynamic Bayesian model of sense change (SCAN). The model infers word sense over continuous time windows as a probability. Thus, capturing the change in Bayesian dependency between adjacent time windows. Word sense change is then represented in the form of probability distribution. SCAN model achieved word meaning change, novel sense detection and word association over different time slices. Although, SCAN model shows significant improvement in capturing word sense change over traditional approaches, however, SCAN method used the bag-of-words as an input that doesn't account for the contextual background the word appears in.

Kulkarni et al. (2014) work included distributional word embedding to detect the time of changing word meaning. Hamilton, Leskovec and Jurafsky (2016)ś paper generates word embeddings by using Word2Vec and employing both the distributional technique and Machine Learning to calculate the changing neighbours for semantic shift of keywords. The results achieved in the experiment showed that temporal semantic shift of words is dictated mostly by the frequency and polysemy. Their findings suggest that more frequent words are less likely to have a shift compared to words that have multiple context.

Gao et al. (2022) proposed a Dynamic Topic Model (DTM), a multistep approach, to study the semantic evolution of topics. The report uses Library and Information Science (LIS) dataset for studying changing semantic of topics. The study followed a multistage experiment that used - a) Topic modelling (LDA) to get probability of word topics. b) Word2Vec for mapping semantics to word and cluster words with similar semantics into same clusters. At the next stage, the study used topic words obtained in step "a", to match the words in semantic clusters from step "b". The key finding of their report showed that most of the topics correspond to different semantic concepts in LIS.

Very recently, Dridi et al. (2022) presented Vec2Dynamics, a novel approach for studying the temporal evolution of keywords in scientific literature. Dridi et al. (2022) employed skip-gram neural network architecture of the Word2Vec. Unlike Word2Vec, skip-gram Word2Vec predicts the surrounding words given the target word. In Vec2Dynamics, the generated embeddings are mapped using K-nearest neighbours, and stability of the k neighbours is calculated to model the changing neighbourhood of keywords over subsequent time windows.

## 2.7 Limitation

The aforementioned approaches have their respective applications. However, bibliometric and hybrid approaches fail to dive into the actual content of the paper and relies on frequencies. Therefore, not capturing the full picture. Content based approaches like Word embedding and distributional techniques account for lexical context of the word by learning a fixed window of words before and after the keyword under consideration Wevers and Koolen (2020). This makes these techniques powerful tools for constructing timeline for semantic shift tracing of words. However, both traditional word embedding techniques (Word2Vec, Glove) and distributional techniques, one way or another, use non-contextualized word embeddings. At most, these techniques take a look at the n-grams. These approaches show significant improvements, but the problem lies in the underlying embeddings employed. These algorithms generate a single vector representation (1-d matrix) for different senses of a word, and may not capture the full context. These discussed approaches may struggle with better representation of the underlying trends, and analysis generated from these word embedding may be prone to errors. Especially in the domains where it's extremely necessary to understand the context (Medical and Finance). As, linguistic is an evolving phenomenon, therefore, the lexical context the word appears in, is as important as the semantic of the word itself, to illustrate the real meaning of the word.

Therefore, this study is an effort to bridge this gap in literature by focusing on this neglected area. The paper aims to generate contextualized word embeddings over successive time windows that machine learning algorithms can be trained on. The methodology of the research is discussed in the chapter 3.

# Chapter 3

# Methodology

This chapter outlays systematic approaches employed to effectively carry out the research project. The project is carried out using qualitative research methodology. The methodology is divided into two sections: Literature Review and Experimental Design Methodology. These approaches are discussed in detail in this section.

## 3.1 Literature Review Methodology

Identification of research gap and building a novel approach, involve critical evaluation of previous literature. Literature Review is the cornerstone of building a solid research foundation. It provides justifications to the research questions and methods chosen to solve the problem statement. Literature Review provides the necessary backing and justification of the chosen research area by pointing the research gap and justification of the chosen design methodology.

For this research, literature review methodology includes:

1. Search Criterion (research questions, searched sources, search terms, time period of the papers and inclusion/exclusion criterion).

2. Selection Criterion (selecting papers, relevance of the paper, paper results, citations and impact).

### 3.1.1 Search Criterion

Sagacious search of the previous research papers is carried in the following databases, Google Scholars, ACM DL, IEEE Explore, Web of Science, ScienceDirect and arXiv. The motivation for choosing these databases is the inclusion of peer reviewed papers that are published in high impact factor journals and A/A* rated conferences. The search terms used are: "Temporal Semantic Shift", "bibliometric analysis", "Diachronic word embedding", "BERT word embedding", "Temporal word embedding" "Deep Learning". Papers from the time period 2010 - 2022 are chosen. The rationale of choosing from 2010 is most of the seminal and pioneering work in the domain of NLP relating to word embedding, contextual natural language embedding and BERT is carried out during this period. The inclusion and exclusion criteria for research papers are given in table 3.1

Table 3.1: Inclusion and Exclusion Criterion of Research Papers

| Inclusion | Exclusion |
|---|---|
| English Papers | Blog articles, Medium, TowardsDataScience |
| Book Chapters | . |
| Papers from 2010 to 2022 | |
| Papers related to linguistics, Science , Machine Learning and Computer Science | |

## 3.2 Experimental Design Methodology

After identification of the research gap and devising the problem statement. The next stage involves the plan of action. The necessary steps undertaken to tackle the problem. This section discusses the plan of actions i.e. tools, techniques to be used to devise the Sci-Bert algorithm.

### 3.2.1 Dataset

For any given machine learning task, acquiring/collecting data is the most crucial part. If the data fed to the algorithm is noisy, then any carefully devised algorithm would give erroneous results. Therefore, the first step is to acquire the data that meets the research criterion and would help in producing and solving the research problem.

As, this paper is an extension of Dridi et al. (2022) study. The study also uses the same dataset as studied in the paper Dridi et al. (2022) for comparative analysis. The Sci-Bert is evaluated in the domain of Machine Learning, with the papers published in Conference on Neural Information Processing Systems (NeurIPS) between the years 1987 and 2017. The dataset used in the analysis includes 6562 conference papers published between the years 1987 and 2017 in NeurIPS. The dataset includes features: timestamps, titles, authors, abstracts, and extracted text. The dataset is available on Kaggle (available at : `https://www.kaggle.com/datasets/benhamner/nips-papers`, accessed 27 May 2022).

### 3.2.2 Sci-Bert

The algorithm Sci-Bert consists of a multi-stage process. In the first step, the dataset comprising papers from 1987-2017 is sliced into 10 time windows. Each time window comprises paper content published during the span of these 3 years. After slicing, preprocessing is applied to each time slice separately to generate a vocabulary set. The vocabulary set consists of unigrams and bigrams of paper content for each time window. The vocabulary set is taken into account, meaning the multiple occurrences of the same bigram or unigram are considered only once.

In order to generate embeddings or vector representation of the words, BERT model was adapted. As briefly described in the previous section, Bidirectional Encoder Representations from Transformers (BERT) was introduced by Jacob Devlin Devlin et al. (2018) and his colleague in the paper first published in 2018. Since itś inception, BERT has illustrated tremendous promise in the area of NLP for contextualized language modelling. It has been extensively used in Google search, Google search Text Summarization, Predictive phrasing in Gmail, capturing order and context of the words.

BERT uses the transformer architecture with stacked encoders that use back propagation and feed forward mechanism to achieve contextualized word embedding. When text is passed to BERT, BERT generates an output matrix of limited size. The matrix contains "CLS" term that maps sentence based embedding as first column, then each respective keyword embedding is represented as subsequent column. The "CLS" embedding part encodes some of the classification (semantic) information of the given sentence.

The benefit of using BERT over traditional technique is, the feature of transfer learning at its core. Transfer Learning has gained popularity in recent years, where models are trained on large corpus and optimized for a variety of use cases. These pretrained models can be fine-tuned by researchers and machine learning practitioners for the domain specific use cases. This helps the training cost and time, making the NLP problems less time-consuming. The idea of fine-tuning, is to use the original model as a base and build on top of it for the specific task.

To facilitate the research and reduce time, the team at Google and Hugging Face have provided many pretrained models of BERT for domain specific tasks. The study adopts this idea of transfer learning. The significance of adapting pretrained model rather than fine-tuned model is to reduce the computation cost and time. Also, the reason for using pretrained model is, fine-tuned models show significantly less performance when trained on small datasets (compared to the pretrained BERT model that is trained on Wikipedia and BooksCorpus). Merchant et al. (2020) points out that fine-tuning primarily affects the top layers of BERT that can lead to large variance of the task performanceDevlin et al. (2019). Also, fine-tuned models suffer from the vanishing gradient problem that can significant reduce performance for unseen context and vocabulary Mosbach, Andriushchenko and Klakow (2020). Furthermore, the rationale is to investigate the role of BERT based embedding over traditional methods in semantic shift tracing. Fine-tuning the model would mean training the additional layers that require both time and computation cost. Although, for the future research, fine-tuning of the model is also under consideration to compare the results of both the pretrained and fine-tuned model.

For the purpose of this report, we use pretrained BERT. The pretrained BERT base model has 12 layers (transformer blocks), 12 attention heads, and 110 million parameters. and is trained on unlabelled data extracted from the BooksCorpus with 800M words and English Wikipedia with 2,500M words (https://en.wikipedia.org/wiki/BERT_language_model, accessed 1 September 2022). BERT comes in two sizes Devlin et al. (2018), BERT base and BERT large. BERT base and BERT large consist of encoder stacks of size 12 and 24 respectively. For the purpose of this report, the algorithm Sci-Bert uses BERT base for generating embeddings. Sci-Bert algorithm is illustrated in detail in the figure 3.1, 3.2,

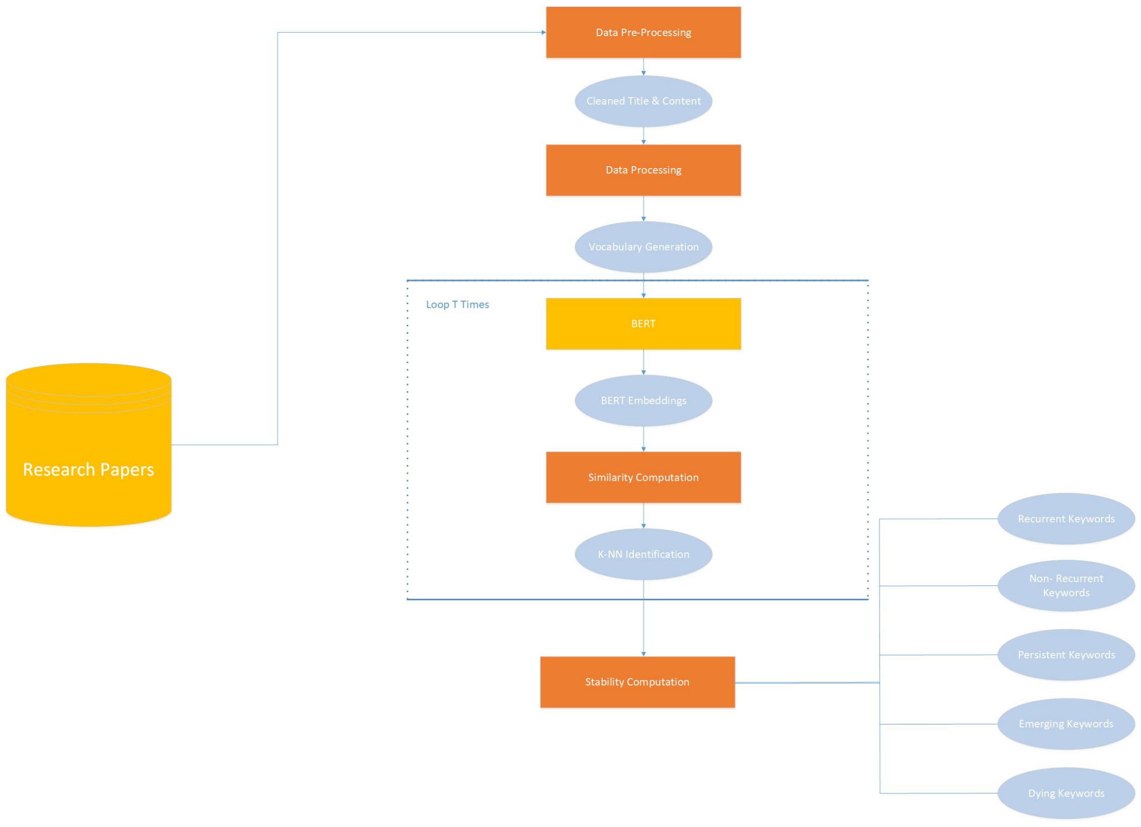


Figure 3.1: Sci-Bert Arhitecture

### 3.2.3 Pre-processing

The first step in any natural language processing task, involves stop words removal. Stop words are the common terms that don't hold much connotational sense and don't hold meaningful significance in the sentence. Stop words include articles, prepositions, pronouns, conjunctions, punctuations etc. For stop word removal, the list "10000 most frequent filtered" was used, (https://github.com/seinecle/Stopwords/blob/master/src/main/java/net/clementlevallois/stopwords/resources/stopwords_10000_most_frequent_filtered.txt, accessed 15 July 2022). The list comprises, 10000 frequent words appearing in Journal and Scientific literature that don't serve meaningful connotation. Some examples include "Fig, Figure, Abstract, en, quote, href, ( )". These words don't hold much semantic significance in determining sentence context. The list is then further combined with the NLTK's stopwords list to get a cleaned corpus for every time slice.

After obtaining the filtered corpus, the regular expression pipeline is applied to the corpus to filter out the words.The regex pipeline consists of a three-step process. During the first phase, words containing digits are removed. Next, words that have a length of 2 or less are pruned out. During the final stage, words that contain any non-alphanumeric characters, like "/", "äre removed from the corpus.

Next, the data is converted to lower case to further narrow down the vocabulary list, removing the difference between uppercase and lowercase words. Programming languages are case-sensitive, therefore, getting a uniform representation for a word is necessary.

During the next phase, word tokens are generated from the pre-processed corpus obtained during the previous phase. Word tokens are generated using the NLTK word_tokeinzer. Each generated word token is then lemmatized i.e. converted into its base form to avoid different versions of the same words as singular, plural, adverbial, verb or noun forms of the same word. We then take the set of the lemmatized tokens to get a unique set of words.

The main goal during the text pre-processing stage is to clean the corpus as much as possible, so the model doesn't take into account, any words that don't have much semantic significance in the scientific literature. This cleaning helps to take out any unwanted words out of the corpus that would impact during the generation of the word embeddings for the candidate keywords. After the whole pre-processing pipeline is executed, the obtained list of words consists of a cleaned vocabulary set. Thus, removing any duplicate versions of the same word.

### 3.2.4 N-Grams

The term n-grams can broadly be defined as, the contiguous substrings of size n from the given sequence of text Järvelin, Järvelin and Järvelin (2007).

Uni-grams are the contiguous substrings where n = 1. Similarly, bigrams and trigrams are the contiguous sequences where n=2 and n=3 respectively. N-gram is a popular tool for Information retrieval (IR).

In this report, N-grams are employed for the purpose of information retrieval. In the case of Sci-Bert, n-grams are employed for the purpose of getting the keywords of interest. Keywords of interest are the top 100-bigrams (n = 1 and n = 2) extracted from the titles of the papers published in NIPS. The titles are recorded under the title column of the dataset. Only the top 100 bigrams are selected that have the most frequency. From these top 100 bigrams, we select only the bigrams that appear in the upper two levels of taxonomy that represent the general terms and categories in the field of Computer Science (`https://skm.kmi.open.ac.uk/cso/`, accessed 5 July 2022). The choice of restricting the bigrams to only 20 is dictated by the rationale of keeping the topics as generic as possible, reflecting the topics rather than the fine-grained analysis of sub-field of the Computer Science domain. The second reason, for keeping the 20 bigrams, is for the comparative analysis with Dridi et al. (2022).

### 3.2.5 BERT

In this report, BERT model was applied in two stages,

1. On the vocabulary extracted from the paper titles.
2. On the vocabulary extracted from the paper content.

Both these stages are discussed in detail in the following sections.

#### Title BERT Embeddings

To apply BERT, two approaches are adapted. First, vocabulary of top 100 unigrams and bigrams are generated from the paper titles. From these unigrams and bigrams, we only keep the top 20 bigrams  unigrams that appear in the highest two levels of the Computer Science Ontology (CSO) (`http://skm.kmi.open.ac.uk/cso/`, accessed 5 July 2022). These two levels reflect the top level of computer science taxonomy and categories. This aligns with our objective to study the topics (keywords) of computer science domain, making the study as generalized as possible, rather than focusing on the fine-grained sub-topics or domains. From now on, these top 20 unigrams and bigrams are referred as "Keyword of interest or Candidate keyword".

Next, for each time window containing the paper content. We only get the sentences that each candidate keyword appears in. These extracted sentences from each time slice are fed to the BERT model iteratively to generate the embeddings list of every candidate list for every time window. The rationale behind using whole sentence is to encapsulate the semantic significance of the sentence the candidate keyword appears in.

The output of the BERT model is a tuple that contains multiple elements, depending on the "Configuration Class"'s parameters used during the model initialization. Generally, the tuple comprises two elements : last_hidden_state and pooler_output. last_hidden_state only gives the output embedding tensors for the last output layer for the given sentence. pooler_output, performs max_pooling and gives the pooled output of the layers of the "CLS" token. The output "CLS" token then can be used for classification tasks and masked language modelling.

For the given use case, the model is configured to output the tuple of size three, with the third element being the "hidden_states". The output of each of the 12 layers (encoder/decoder) stack of the Bert_base is stored in these "hidden_states". As the sentence passes through each layer of the stack, these hidden states hold the output of each layer. The objective of this research is to encapsulate the semantics for each of the candidate keyword. To encode semantic information of the given word in context to the sentence, BERT uses self-attention. Self-attention computes each token against the other input sequences in the sentence for better semantic clarity and association of the given word. "hidden states" is of size 12, storing 12 output embeddings of each stack for a given input.

The hyperparameters used in the experiment were; "hidden_size"=768, representing the output embedding length for each input. num_hidden_layers=12, that showcase the number of transformers stack in BERT. num_attention_heads=12, Number of parallel attention layers for computing self-attention. intermediate_size=3072, represent the size of feed-forward layer that is applied to respective token position. hidden_act="gelu", Gaussian Error Linear Unit (GELU), the activation function used in the model. max_position_embeddings=512, vector dimensions to encode the positional information of the token. max_position_embeddings can be of size (512, 1024, 2048). We went with 512 that was used by Devlin et al. (2018) for the BERT_base model to achieve state-of-the-art results. In fact, the choice of hyperparameter is influenced by the original paper Devlin et al. (2018).

In this research, the last 4 outputs of the "hidden states" for the candidate keywords are used. In BERT, for each word token, Every layer does multi-headed attention computation on the word representation of the previous layer to create a new intermediate representation. The rationale on using the last 4 layers is, in case even if some of the semantic meaning is lost during the final layer, the summed output of the last four layers for each candidate keyword ensure the encapsulation of the original semantic context.

The previous stage, for the candidate keywords, is repeated i times, where $i = NumberofTimewindows$. The whole BERT embedding generation process is illustrated in figure 3.2

Paper Content BERT Embeddings

During the next stage, vocabulary is generated for each time window consisting of the contents of the paper during that time period. Vocabulary of each time window consists of unigrams  bigrams extracted from the vocabulary. To keep the words concise and as meaningful as possible, First, the vocabulary consisting of less than 2 characters was excluded from the list. Then, only the vocabulary words that had significance presence in the corpus were included in the vocabulary set. For this, the words that had frequency of occurrence less than 5 were not considered. The reason for excluding such words was, as these words don't appear much in the corpus, this would help in pruning out words that don't hold much significant importance in the domain/field of study (here, it is ML or Computer Science (CS)). These words may include, author names, or terms that aren't frequent in literature.

After the vocabulary set is generated, the methodology in this step is the same as in section 3.2.5 discussed. First, We go over each time slice, For every word appearing in the vocabulary during the given time slice, we get the list of sentences from the paper content

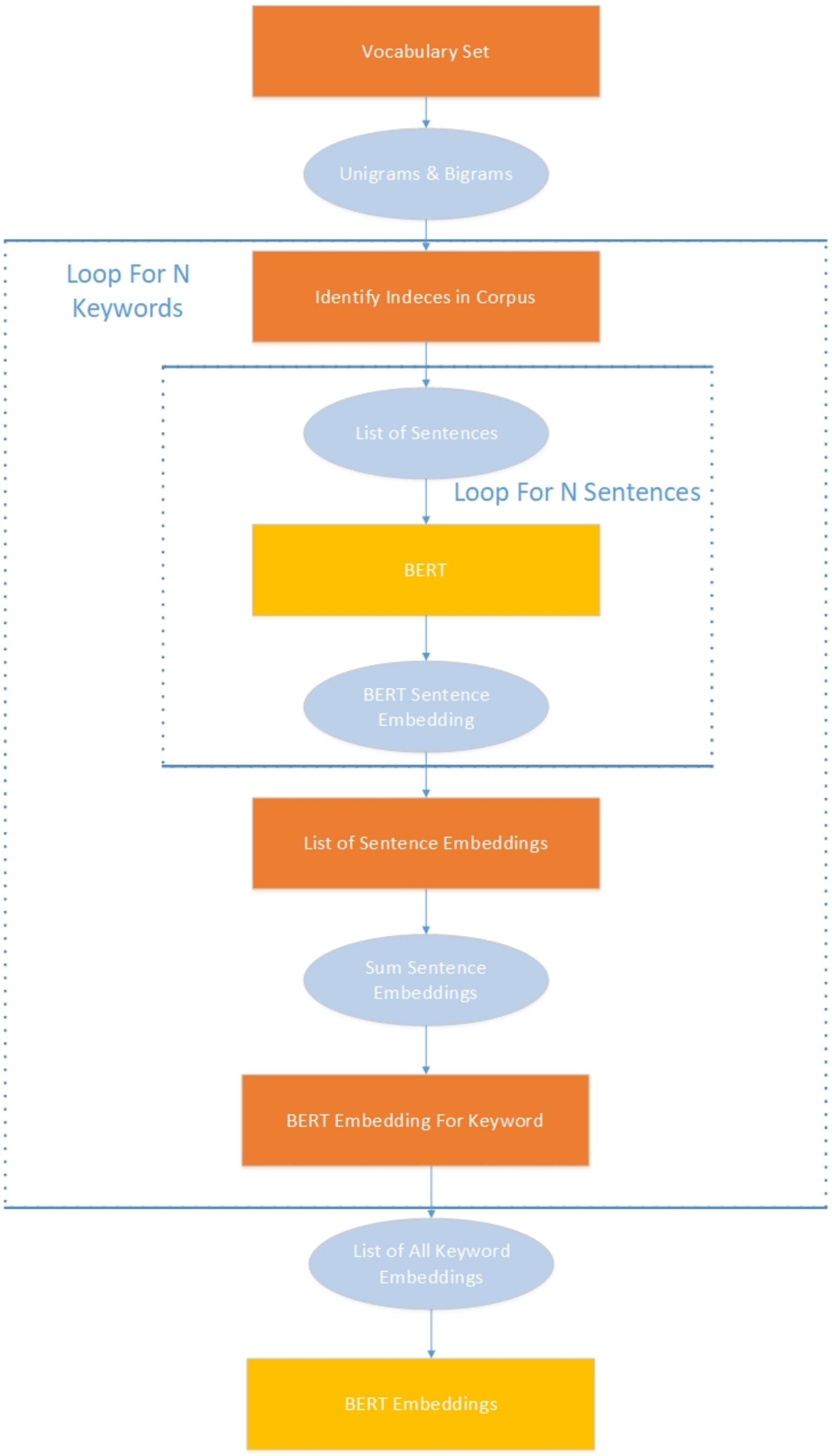


Figure 3.2: BERT Embedding Generation Process From Vocabulary Set and Candidate Keywords

during that time period.

Next, after getting the list of sentences for every word in the vocabulary. The sentences are passed to BERT model for embedding generation. This helps to generate the embeddings that are contextualized and capture the semantic content of the sentences in which the word appears. Lastly, the last 4 hidden state token embeddings are considered and taken into account.

### 3.2.6 Similarity Measure

After getting the embeddings for both the candidate keywords and corpus vocabulary. Generated embeddings for every time slices are then evaluated based on the cosine similarity score.

When dealing with vector spaces, it's important to preserve the intrinsic characteristic of the vectors in the embedding space. To do so, There are different distance measures for checking the similarity of vectors, i.e. how similar two vectors are to each other. It's critically important in natural language tasks where the words are represented in the form of number, or more formally in the form of matrices. For this reason, there are different distance measures to compute the similarity of vectors, i.e. Euclidean, Dot Product and cosine similarity. For the purpose of the given research, we applied Cosine Similarity and Dot product similarity. Although, we apply both similarity measures, but we focus more on the results of cosine similarity due to the objectives of this report. Also, For text analysis, it's frequently used to gauge the similarity of the documents Han, Kamber and Pei (2012).

Furthermore, The rationale behind focusing more on cosine similarity is, we're dealing with geometric space and are only concerned about the angles between semantically similar vectors. Dot product similarity could also be employed as the main similarity measure, however, as we're concerned with only the angles and not the magnitude. Also, scaling the data would just make the two metrics similar. Although, cosine similarly is employed for this experimentation. But for checking the results using different measures, Dot product was also used to see which similarity metric gave better results.

Mathematically, cosine similarity can be defined as the cosine angle between of two vectors in hyper or multidimensional space Gupta, Sachdeva and Dohare (2021). Cosine similarity between two vectors is defined as;

$$similarity(x, y) = \cos(\theta) = \frac{x.y}{|x|\,|y|} \tag{3.1}$$

where $x.y$ represent the dot product of the two vectors. $|x|$ is the norm of the vector or the magnitude of the vector. The angle $\theta$ determines how similar or dissimilar the two vectors are. If the angle between the vectors is $90°$, when the vectors are orthogonal, then the cosine similarity value would be 0. Highlighting no similarity between the two vectors. Closer the value of cosine similarity to 0, smaller the angle between the vectors and more identical the two vectors to each other.

### 3.2.7 K Nearest Neighbors (K-NN)

In the next stage, K-nearest neighbour (K-NN) is employed. The similarity scores obtained at the previous stage are then used to generate the K-nearest neighbours of candidate keywords for each time window. Here, nearest neighbours correspond to the vocabulary words appearing during that time window. Here k=10, the choice of selecting k=10 is dictated by two factors. 1) for keeping the methodology same as Dridi et al. (2022). 2) by the study conducted by Hall, Jurafsky and Manning (2008) on the history of trends in computational linguistics. Their findings suggest that the set of 10 words was successful to describe each topic. The choice of using K-NN is dictated by the data being unlabelled and lack of prior metrics that quantify the evolution of Computer Science words. Although, here, K-NN is used but any unsupervised machine learning algorithm can be employed. As the project was time contained, in future, other machine learning models like DBSCAN and Chameleon models are also under consideration for result comparison of these algorithms in tracing semantic shift over time.

### 3.2.8 Stability Calculation & Venn Diagrams

Next, the stability of K-NN for each candidate keyword is computed over time. Let $S^{t1}_{w_i}$ and $S^{t}_{w_i}$ denote the sets of k-NN of the keyword $w_i$ over two successive time windows $t1$ and, $t$ respectively. $\psi_t$ denotes the K-NN stability of $w_i$ at time window t as the logarithmic ratio between the intersection of two sets $S^{t1}_{w_i}$ and $S^{t}_{w_i}$, denoting the vocabulary words for the candidate keyword $w_i$, divided by the symmetric difference between the two sets. Mathematically, Stability of K-NN is defined by the equation;

$$\psi^t_{w_i} = \log_k\left(\frac{\left|S^t_{w_i} \cap S^{t-1}_{w_i}\right|}{0.5 \times \left|S^t_{w_i} \ominus S^{t-1}_{w_i}\right|}\right) \tag{3.2}$$

Or

$$\psi^t_{w_i} = \begin{cases} -1 \log_k(\frac{1}{k}) & \text{if} \quad \mathrm{S}^t_{w_i} \cap S^{t-1}_{w_i} = \emptyset \\ \log_k(k) & \text{if} \quad \mathrm{S}^t_{w_i} = S^{t-1}_{w_i} = k \\ otherwise & \quad \text{Equation 3.2} \end{cases} \tag{3.3}$$

Stability of a candidate keyword $w_i$ over two successive time periods is defined as; intersection of nearest neighbour keywords of two successive time windows, divided by the symmetric difference of the two sets multiplied by 0.5, and taking the log of the values.

We multiply the denominator by 0.5 because, we take the symmetric difference between the two sets. As, both sets are symmetric, meaning having equal set elements, thus making the denominator equal to or less than the number of chosen k value.

The value of $\psi_t$ ranges between -1 and 1. We compute stability by $-1 \log_k(\frac{1}{k})$ If there is no intersection between the two sets, meaning $S_{w_i}^{t} \cap S_{w_i}^{t-1} = 0$ to avoid having the numerator equal to 0. Similarly, we refer to the equation $\log_{k}(k)$ , if both sets have same elements i.e. $S_{w_i}^{t} = S_{w_i}^{t-1} = k$ to avoid having denominator equal to 0. For every other case, stability of the keyword is measured using the equation 3.2.

After computing the stability for each candidate keyword $w_i$ over all time windows T. Overall, Average stability $\psi^t$ is calculated for the n selected candidate keywords for each time window t. Average stability is given as follows;

$$\psi^t = \frac{\sum_{n-1}^{n} \psi_{w_i}^{t}}{n} \tag{3.4}$$

The stability of the candidate keywords highlight the dynamism of the keyword $w_i$ over two successive time windows t-1 and t. Large intersection between the two time windows suggest the field / domain $(w_i)$ to be stationary during that time period.

Also, the words in the vocabulary set of the candidate keyword $w_i$ can be classified into five different categories based on the dynamism of the words: recurrent keywords, non-recurrent keywords, persistent keywords, emerging keywords and dying keywords. These keyword categories are defined in detail in Dridi et al. (2022) and are briefly explained below:

Definition 1 (Recurrent Keywords): A word is considered a recurrent keyword if it appears in more 2 or more successive time windows.

Definition 2 (Non-Recurrent Keywords): A word is considered non-recurrent if it appears in more 2 or more time windows but not in succession.

Definition 3 (Persistent Keywords): A word is considered persistent if it appears in two consecutive time windows.

Definition 4 (Emerging Keywords): A word is regarded as emergent if it isn't present in previous time window but emerges in the next time window.

Definition 5 (Dying Keywords): A word is regarded as dying if it is present in the previous time window but isn't present in the next time window.

Lastly, a comparative analysis with stability results of the paper Dridi et al. (2022) would help to compare the impact of BERT based embedding approach over word2vec.

# Chapter 4

# Experiments

To track the dynamics of the keyword in temporal fashion for the domain of Machine Learning, NIPS dataset (https://www.kaggle.com/datasets/benhamner/nips-papers accessed 27 May 2022) has been chosen. The Conference and Workshop on Neural Information Processing Systems (abbreviated as NeurIPS and formerly NIPS) is a machine learning conference and is ranked $2^{nd}$ in the field of machine learning after IEEE (Computer Vision and Pattern Recognition) that is a more domain specific in the area of computer vision (https://research.com/conference-rankings/computer-science/machine-learning accessed 15 August 2022).

NIPS is a more generalized conference, covering a wide range of topics from machine learning i.e. computer vision, deep learning, knowledge discovery, data mining, big data and I.O.T. This aligns perfectly with the objectives of this research that requires a more descriptive analysis of the papers in temporal fashion. For the analysis, NIPS dataset is also highly publicly available 3.2.1, is managed and curated with time stamped papers.

This chapter is divided into two sections; the first section discusses the experimental setup and dataset statistics. The second section elaborates on the experimental results and key findings obtained after employing the Sci-Bert algorithm.

## 4.1 NIPS Dataset

NIPS dataset was used for the analysis of Sci-Bert. The dataset is available on Kaggle (https://www.kaggle.com/benhamner/nips-2015-papers/data, accessed 27 May 2022). The NIPS data contains "Papers", "Authors" and "PaperAuthors" datasets. For this report, "Papers" data was used.

"Papers" data contains 7241 papers published from the years 1987-2017. The dataset consists of features: Id, Title, EventType i.e. (Poster, Spotlight and 15 other categories), PDF name, Abstract and PaperText.

The dataset was first pre-processed following the methods discussed in the section 3.2.3. The dataset was then sliced into 3 years time windows to check the dynamism of the candidate keywords using Sci-Bert. We only take into consideration papers from 1987-2016 to keep the time period size of 3 across all time windows. Therefore, the papers published during the year 2017 aren't part of the analysis. The choice of time window being the size of 3 is inspired by the methodology adopted by (Dridi et al., 2022; Fremann and Lapata, 2016), Anderson, Jurafsky and McFarland (2012). The rationale is to keep the methodology as similar as possible for result insights. The statistics of the published papers is given in the table 4.1:

Table 4.1: NIPS Dataset Statistics

| Time Window | Papers | Words | Vocabulary |
|---|---|---|---|
| 1987 - 1989 | 285 | 846216 | 5922 |
| 1990 - 1992 | 414 | 1152723 | 8102 |
| 1993 - 1995 | 450 | 1328025 | 9386 |
| 1996 - 1998 | 453 | 1391303 | 9496 |
| 1999 - 2001 | 499 | 1629734 | 10602 |
| 2002 - 2004 | 612 | 2325949 | 14576 |
| 2005 - 2007 | 628 | 2707689 | 16917 |
| 2008 - 2010 | 804 | 3960937 | 25292 |
| 2011 - 2013 | 1032 | 5356559 | 35149 |
| 2014 - 2016 | 1383 | 7263591 | 49693 |

The table 4.1 highlights the key statistics about the "Papers" dataset. It is evident from the table that there is a general upward trend in the number of papers published from 1987 to 2016. It is interesting to know that the number of published papers significantly increase during the years 2008 to 2016 from 804 to 1383 that's a 58% increase. This, in turn, also increases the word count in the papers and the generated vocabulary significantly.

"Words" column highlights the total number of words present in all the papers during the given time window. The general trend of increase in published paper is also reflected in the word count.

"Vocabulary" column highlights the vocabulary (unigram and bigrams) that was taken into account during the given time window for the Sci-Bert algorithm. Here, vocabulary size highlights the words that had word frequency of more than 5 and were not stopwords.

Sci-Bert algorithm is applied independently on all the time windows separately. This also helps to gauge the performance of Sci-Bert, especially BERT based embedding in identifying similar topics to the studied candidate keywords, as each time window can be regarded as a sub-dataset.

## 4.2 Results and Discussion

The dynamics of the keywords studied in this paper are tracked in a temporal fashion from the years 1986 to 2016 on NIPS paper. For this purpose, we applied BERT based embedding that takes the context of each sentence, the keyword from vocabulary set appears in.

After preprocessing discussed in section 3.2.3, we divided papers into their respective time slice depending on the time stamp of the paper. After time slice, a vocabulary set was generated for each corpus. Then we tracked every sentence that each word from the vocabulary set appears in and applied BERT on the obtained sentences of the word to generate contextualized word embeddings. The hyperparameters used for BERT embedding were, the vector dimension of size 768, attention vector of size 512 and hidden layers of size 12. The hyperparameter choice were influenced by the seminal work Devlin et al. (2018).

We generate the BERT embeddings in two phases, First, we generate BERT embeddings for the candidate keywords (unigrams and bigrams). The candidate keywords are our keywords of interest that correspond to the top 100 bigrams that we generated from the title of the papers from all time windows. These candidate keywords highlight the key area or domains of machine learning. From these top 100 candidate keywords, we only kept the candidate keywords studied in Dridi et al. (2022) paper and that appear in the top two levels of CSO taxonomy (http://skm.kmi.open.ac.uk/cso/, accessed on 5 July 2022).

During the second phase, we applied BERT embeddings to each word in our vocabulary set generated from the paper contents of the dataset. Similarly, like the first phase, we keep track of all sentences the word appears in and apply BERT embeddings to the tracked sentences.

After BERT embeddings are generated for both the Candidate keywords and Vocabulary set, we apply cosine similarity given by equation 3.1 and generate K-NN. "K" is kept to 10, i.e. we track the 10 nearest words for each our candidate keyword. To be more precise, we generate 300 K-nearest neighbours for each candidate keyword returned by cosine similarity. From these 300 keywords, only those keywords are selected that appear in the top two levels of CSO taxonomy. In fact, different values of k were chosen, but the value of $k = 300$ ensured the availability of 10 nearest neighbours that appear in the top two levels of CSO taxonomy.

After getting the K-nearest neighbours for each of the candidate keyword. The stability of the candidate keyword was calculated in temporal fashion, described by the equation 3.2. Also, the average, stability of each time span was also calculated using the equation 3.4. We omitted the average stability of the first time overlap between the time windows 1987-1989 and 1990-1992, for keeping the methodology consistent with Dridi et al. (2022).

K-NN stability of the top 20 keywords studied in this paper are shown in the table 4.2;

Table 4.2: K-NN Stability of top 20 Candidate keywords studied

| Keywords | 93-95 | 96-98 | 99-01 | 02-04 | 05-07 | 08-10 | 11-13 | 14-16 |
|---|---|---|---|---|---|---|---|---|
| monte carlo | 0.176 | 0.368 | 0.368 | -0.368 | 0.176 | 0.602 | -1.0 | 0.368 |
| gradient descent | 0.176 | 0.368 | 0.176 | -0.602 | 0.0 | 0.176 | -1.0 | -0.243 |
| supervised | 0.0 | 0.176 | 0.602 | -0.602 | 0.368 | 0.368 | -1.0 | 0.368 |
| reinforcement learning | 0.176 | 0.0 | 0.176 | -1.0 | 0.176 | 0.368 | -0.954 | 0.602 |
| guassian | 0.176 | 0.176 | 0.368 | -0.602 | 0.176 | -0.176 | -1.0 | 0.38 |
| graphical model | 0.0 | 0.0 | 0.0 | -1.0 | 0.0 | 0.368 | -1.0 | 0.954 |
| dynamic programming | 0.176 | 0.176 | 0.602 | -1.0 | 0.0 | 0.0 | -1.0 | 0.368 |
| neural network | 0.0 | -0.176 | -0.176 | -1.0 | 0.176 | 0.368 | 0.0 | 0.602 |
| time series | 0.176 | 0.368 | 0.0 | 0.602 | 0.0 | 0.368 | 0.176 | 0.368 |
| deep learning | -1.0 | -1.0 | 0.0 | -1.0 | 0.368 | 0.368 | -1.0 | 0.602 |
| machine learning | 0.602 | 0.176 | 0.602 | -0.954 | 0.176 | 0.602 | -1.0 | 0.602 |
| markov | -1.0 | -1.0 | -1.0 | -0.602 | 0.368 | 0.602 | -0.954 | 0.368 |
| supervised learning | 0.176 | 0.176 | 0.176 | -0.954 | 0.176 | 0.602 | -0.954 | 0.954 |
| artificial neural | 0.368 | 0.368 | 0.602 | -1.0 | 0.602 | 0.602 | -1.0 | 0.368 |
| component analysis | 0.602 | 0.368 | 0.368 | -1.0 | 0.176 | 0.954 | -0.954 | 0.368 |
| nearest neighbor | -1.0 | -1.0 | -1.0 | -1.0 | 0.176 | 0.368 | -1.0 | 0.368 |
| gradient | 0.0 | 0.176 | 0.176 | -1.0 | 0.368 | 0.0 | -1.0 | 0.368 |
| model | 0.176 | 0.176 | 0.368 | -1.0 | 0.368 | 0.602 | -1.0 | 0.0 |
| learning | 0.602 | 0.368 | 0.0 | -1.0 | -1.0 | 0.0 | -1.0 | 0.602 |
| Average Stability | -0.0475 | -0.016 | -0.031 | -0.0163 | -0.0839 | -0.0339 | 0.0809 | 0.09 |

The table 4.2 illustrate, the stability of each candidate keyword studied. The average stability of each time span is shown at the bottom of the table. The average stability of all the time windows range between -0.0839 to 0.09. A better visualization of the average stability over time is illustrated in the fig 4.1.

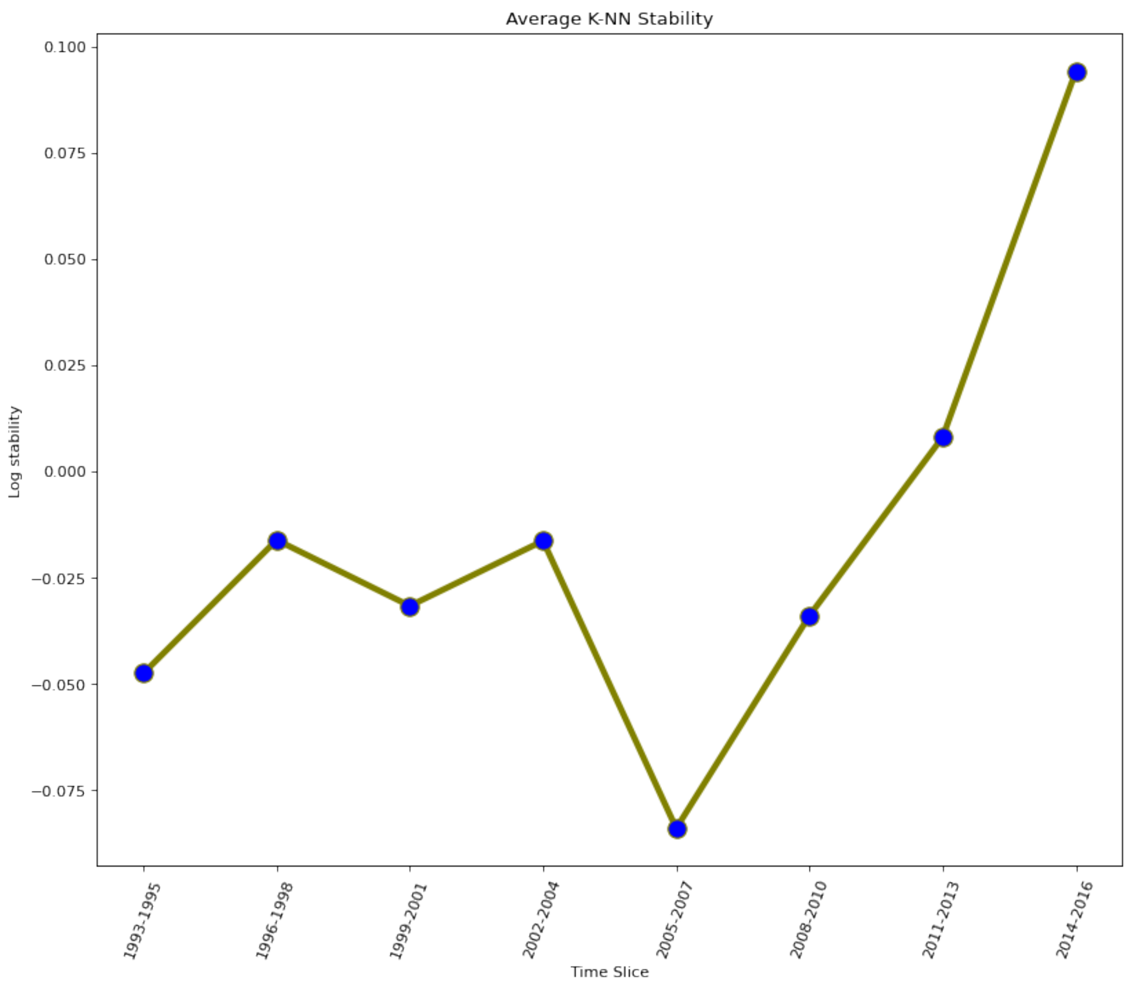


Figure 4.1: Average Stability of each time span

Figure 4.1 highlight the average stability in temporal fashion. The average stability is generally negative, ranging between -0.08 and 0.09 suggesting disruption in the field.

The figure 4.1 shows a general trend of low stability during the 90s and early 2000. The two of the lowest stabilities were detected during the years 1993-1995 and 2005-2007. The low stability during the year 1993-1995 is quite interesting, as it highlights that the field has been more receptive of new ideas and innovations during these years. This is, in fact, suggested by the timeline of machine learning (`https://timelines.issarice.com/wiki/Timeline_of_machine_learning` Accessed 07 September 2022), with the discoveries of Random Forest Algorithm Breiman (2001), Support Vector Machines Cortes and Vapnik (1995).

From the years, 1996-1998, the stability is still quite low but increased a bit from the previous time window, This is supported by the timeline earlier, as during this time the only prominent discovery was LSTM Hochreiter and Schmidhuber (1997) during the year 1997. The other major events during this time were IBM Deep blue beating the word champion at chess and the release of MNIST database by the team led by Yann LeCun that became a benchmark for handwriting recognition. Although, LSTM was a significant discovery, however, MNIST is a dataset and is difficult to regard as discovery. That definitely shows the improved stability from the previous time window.

From the years 1999-2001, we again see a slight drop in stability, pointing out to some discoveries in the field. This is further supported by the machine learning timeline, during this time many machine learning algorithms were developed. During this time frame, LogitBoost Friedman, Hastie and Tibshirani (2000) that is an extension of AdaBoost (the related words of AdaBoost appear in overlapping neighbours of other candidate keywords in Sci-Bert Analysis), anomaly detection, the local outlier factor Breunig et al. (2000) were the algorithms proposed during this time frame. Computer-aided cancer diagnosis prototype was developed by University of Chicago, that detected cancer 52% more accurately than radiologists.

Despite still being low, we see a slight improvement in the stability during the time window 2002-2004, the only notable algorithm introduced during this time was, manifold alignment. The concept of manifold alignment was first introduced in the year 2002. Other than that, Google introduced MapReduce framework in 2004 for parallel processing and computing Dean and Ghemawat (2004). Google MapReduce is regarded as technology and still a lot of literature is put out using the framework MapReduce.

During the time window 2005-2007, we see the lowest stability. It is in-fact supported by the timeline of machine learning, as 2005 is regarded as $3^{rd}$ rise of machine learning where the discoveries of the past and present by Hinton, LeCun, Bengio, Andrew Ng and others started taking shape in conjunction. The term "Big Data" was first introduced during the year 2005, followed by the term "deep learning" coined by Geoffrey Hinton In 2006 ushering in a new era of deep learning research LeCun, Bengio and Hinton (2015). In 2006, Netflix announced the competition to beat its recommender system. In 2006, Face Recognition Grand Challenge was organized for mapping 3D facial recognition. Their findings suggested newer algorithms to be more suitable and better in performance compared to the algorithms of 2002 and earlier.

From the years 2009 to 2017, we see increase in stability. Increase in stability suggest that not a lot of ground-breaking discoveries (algorithms) are made during these years. In fact, this is true, highlighted by the machine learning timeline. To be more precise, most of the new work during this time frame is on the open source and commercialization of machine learning applications. Different machine learning libraries like Pandas, Theano, spaCy, Apache Mahout, Apache Spark, Keras, Pytorch, Tensorflow were released. During this time frame, few algorithms oriented research like DeepFace by Facebook Taigman et al. (2014), AlexNet Krizhevsky, Sutskever and Hinton (2012) that achieved state-of-the-art result in ImageNet competition by first introducing ReLU activation function for CNNs, Google's X Lab team developed a machine learning algorithm that identifies images of cat by watching unlabelled images.

To further illustrate, we created the Venn Diagrams of the K-NNs of the keyword "machine learning" to illustrate the timeline and studies over the 30-year time frame.

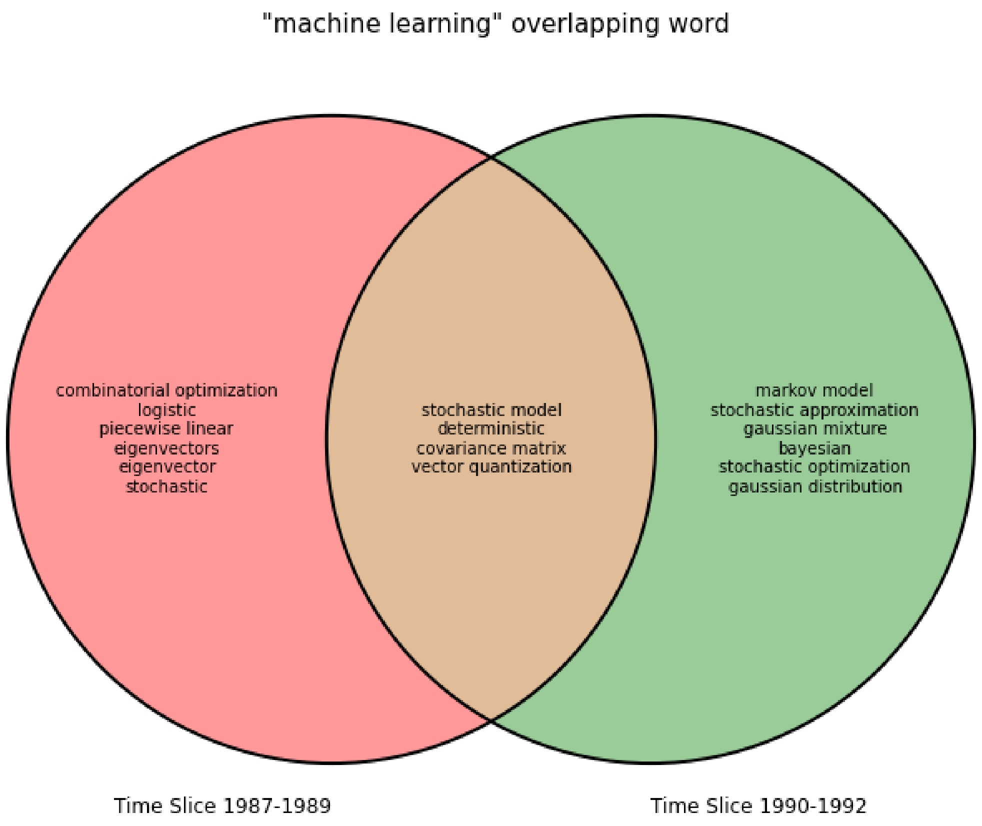


Figure 4.2: Overlapping Topics (words) for "Machine Learning" keyword For Timeline 1987-1989 and 1990-1992

As illustrated from the figures 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10, BERT was able to map the history of keyword "machine learning" from the years 1987 - 2016. The interesting thing about figures is, the results highlight the areas of research undertaken in the area of machine learning. If we have a look at the time windows 1987-1989 to 1990-1992, there are few overlaps. Compared to the time frame 1990-1992 to 1993-1995. Here, the word "stochastic optimization" and "markov models" can be termed as "Recurrent Keywords", appearing in 3 time windows respectively. The word "tensor factorization" can be termed as "Dying keyword" as it appears in time window 2008-2010 but not in the next time window. The definitions for multinomial classification of words is detailed in the section 3.2.1.

Another interesting finding from the algorithm, highlights that from the years 2008-2010 and 2011-2013, There are not a lot of overlaps. Many new topics like "Kalman Filter", "wavelet transforms", "Laplacian Eigenmaps", "Lagrange multiplier" are the topic of research. If we dissect the topics, we would notice most of these topics are studied under the area of "Deep learning" or to be more precise, these topics are studied in "Computer

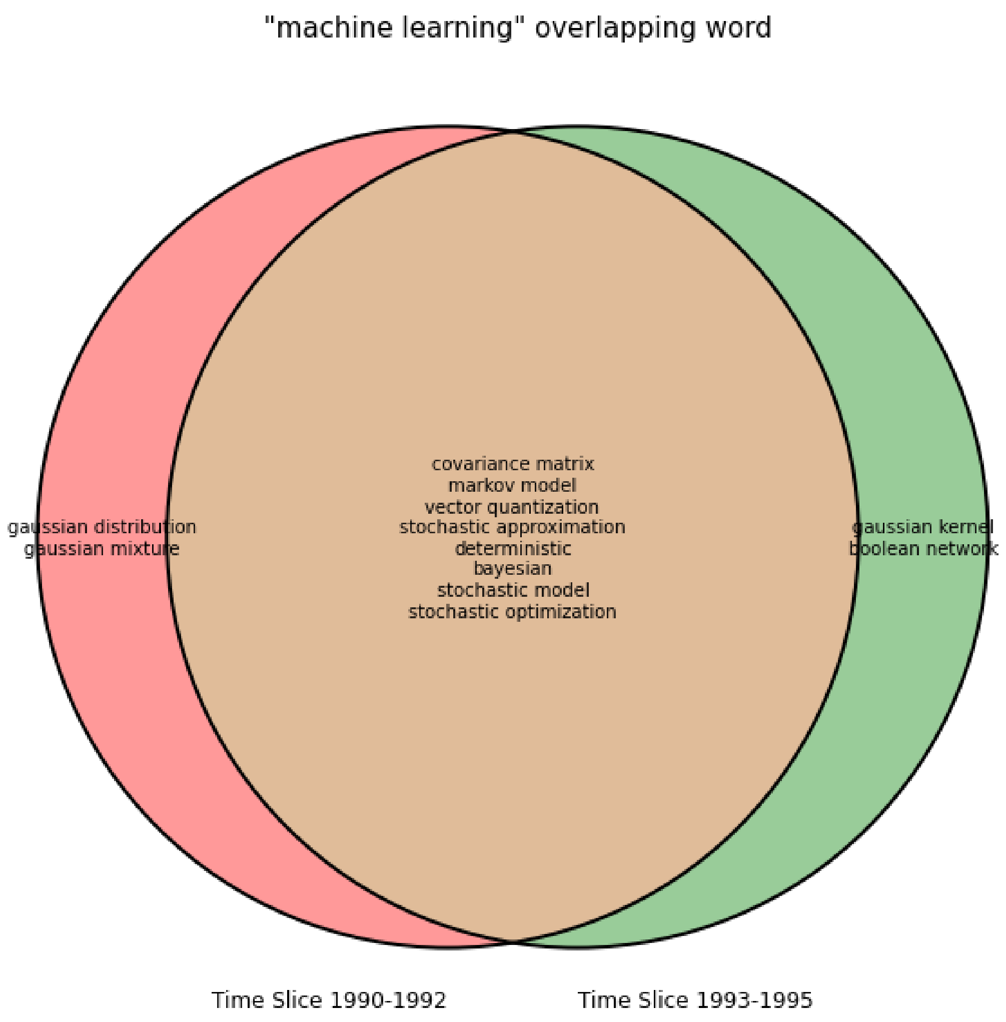


Figure 4.3: Overlapping Topics (words) for "Machine Learning" keyword for timeline 1990-1992 and 1993-1995

Vision", "Image processing","Robotics", "Language Modeling" and "Dimensionality Reduction". All of these techniques are the cornerstone for matrix and vector optimizations. Another interesting thing to note here is, techniques like "Gaussian filter", "Gaussian distribution", "Gaussian noise", "Gaussian kernel", "Gaussian model" are the techniques that are studied in the area of image processing. This is further evidenced by the papers and research done in the area of Computer Vision Krizhevsky, Sutskever and Hinton (2012), Taigman et al. (2014).

The Venn diagrams show a clear shift of machine learning paradigm from algorithmic discovery and the study of neural networks to more of an area of optimization and application in the area of deep learning, specifically in the area of computer vision and natural language processing.

In addition to the Cosine Similarity, we also applied Dot Product similarity to investigate which similarity measure yielded better results. The results of the Dot Product similarity

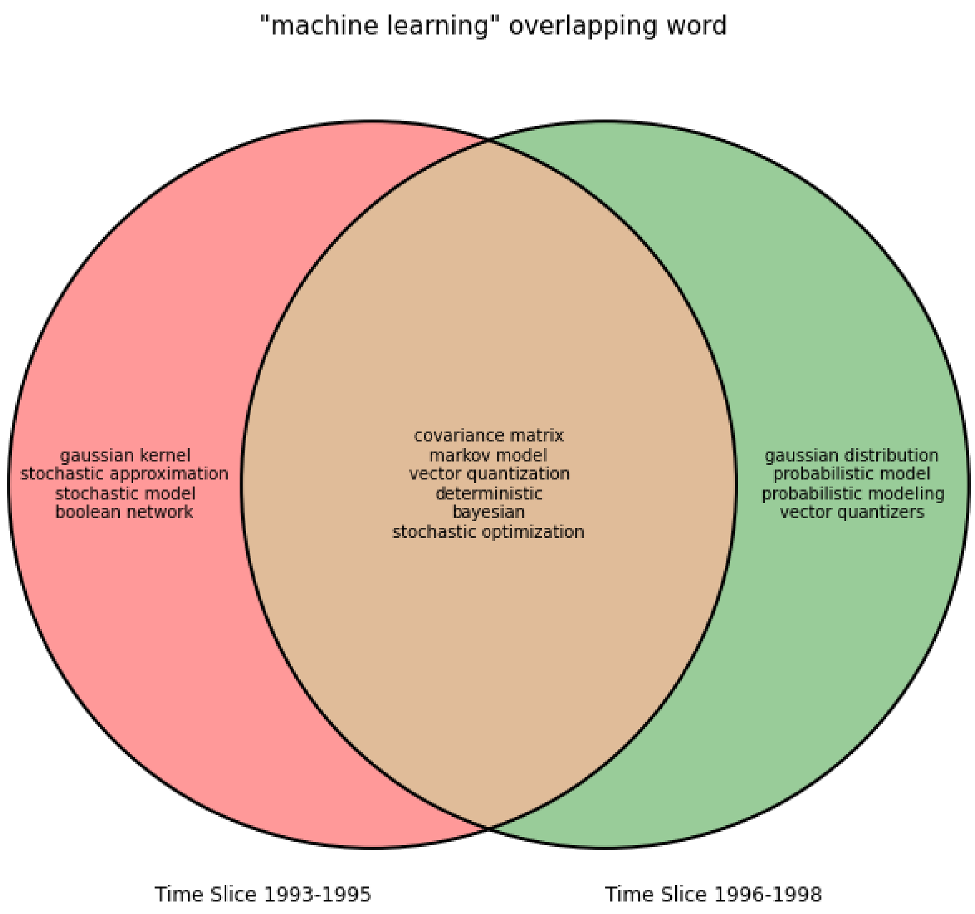


Figure 4.4: Overlapping Topics (words) for "Machine Learning" keyword for timeline 1993-1995 and 1996-1998

are available at appendix B. Dot product similarity struggled to identify the similar words for the "machine learning" domain. In the majority of the time windows, Dot product similarity couldn't identify the keywords that appear in the top two levels of CSO taxonomy. This suggests a very intuitive understanding of how both the similarity measures work. In our research, we applied BERT to generate embeddings of size 768. While cosine similarity only concerns with the angles, i.e. if the vectors point to the same direction in vector space. On the other hand, dot product also accounts for the vector length.

For example, let's take a look at two example sentences. "Sci-Bert report is amazing and absolutely thoughtful. Amazing!". and "Sci-Bert report is written with amazing detail. Not only is it amazing, but the quality of content is thoughtful. The diagrams help evoke emotions in an amazing manner. Absolutely thoughtful". In the first example sentence, Frequency of word "Amazing" = 2 and "Thoughtful" = 1. In the second sentence, "Amazing" = 4 and "Thoughtful" = 2. Although, both sentences (documents), imply same meaning but have different frequency. If we calculate the cosine similarity, the similarity value would be "1", suggesting both vectors (documents) to be the same. To be more precise, if we scale the data then the dot product becomes the cosine similarity as we are

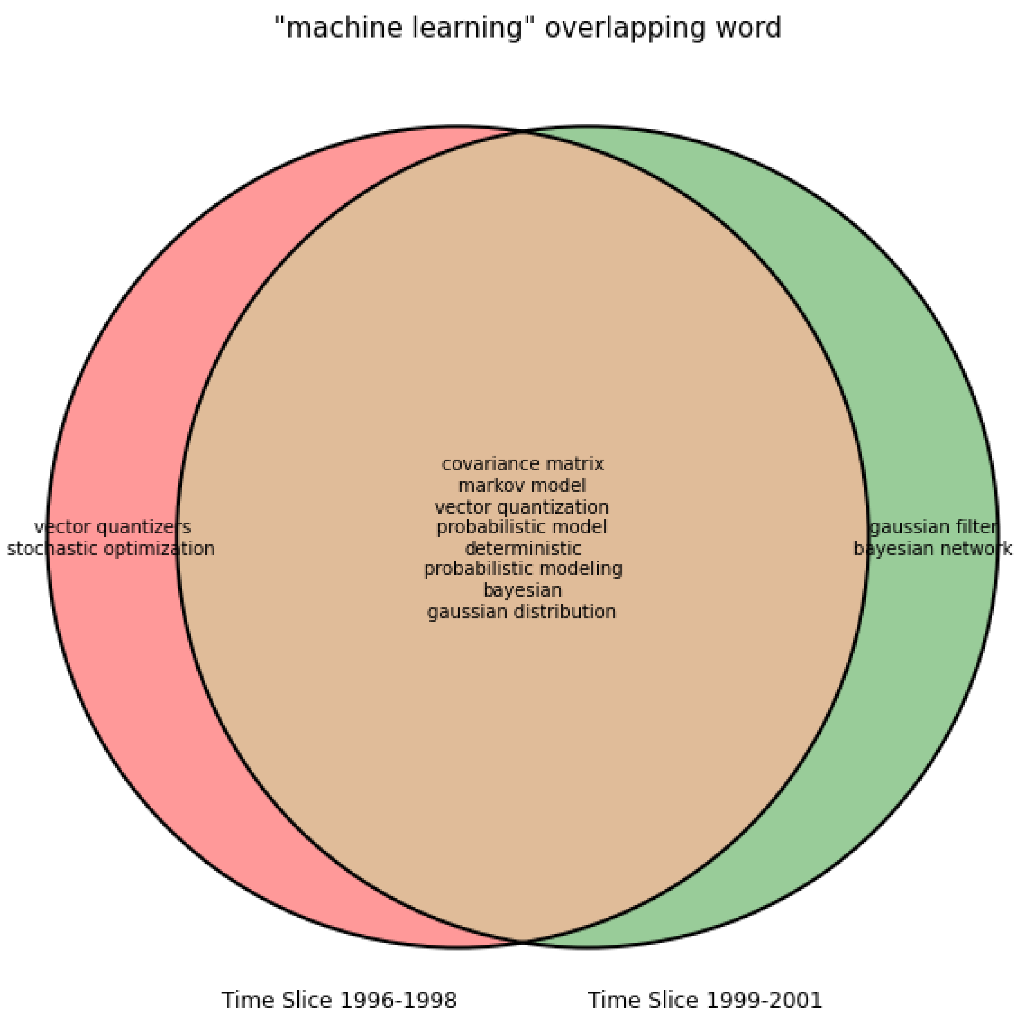


Figure 4.5: Overlapping Topics (words) for "Machine Learning" keyword for timeline 1996-1998

not concerned with the length of the vectors.

In the case of Sci-Bert, as the contextual embeddings were the input for our similarity measure, so, we were more concerned on the semantic similarity of the documents rather than the frequency or length of the vectors. In case, our vector embeddings involved, techniques like TF-IDF, Bag of Words, it would have made sense to use Dot Product over Cosine Similarity. However, in relation to Sci-Bert, cosine similarity produced remarkable results and performed better than Dot product in identifying the K-Nearest Neighbors of the candidate keywords, which is not very surprising, considering how both similarity measures work.

### 4.2.1 Comparison

Figure 4.11 and 4.1 showcase, the average stability of both Vec2Dyanimcs Dridi et al. (2022) and Sci-Bert respectively. If we take a look at both graphs side-by-side. Both

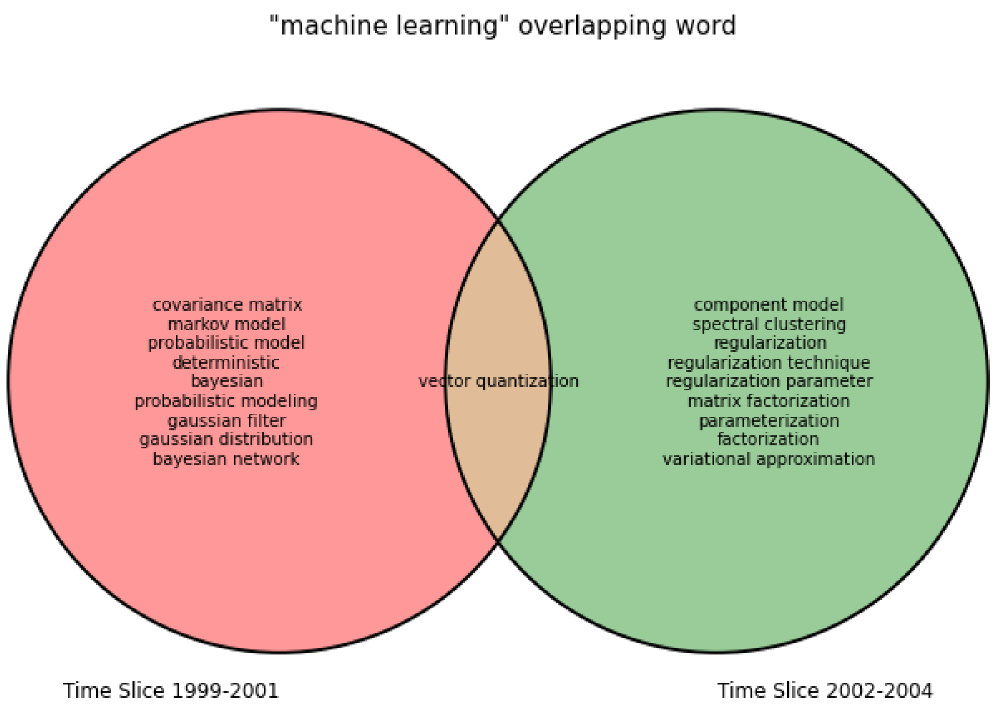


Figure 4.6: Overlapping Topics (words) for "Machine Learning" keyword for timeline 1999-2001 and 2002-2005

graphs depict very similar pictures in terms of stability. For both Vec2Dynamics and Sci-Bert, we get very low stability over the whole timeline. For both studies, the stability remains negative for the majority of the timeline.

Furthermore, the stability fluctuates from years 1993 to 2010 for both Vec2dynamics and Sci-Bert. Only in the last time window, the stability increases in case of Vec2dynamics, suggesting that the field has become more stable. In case of Sci-Bert, The last two time windows show stability. This points out to a more stable domain of machine learning, as can be tracked from the history of machine learning timeline and Venn diagrams that show more research into optimizations rather than introduction of new groundbreaking discoveries.

Although the graph for both Vec2dyanmics and Sci-Bert are pretty much identical. However, it's worth pointing out that the range of stability values in both Vec2dyanmis and Sci-Bert are different. This can be attributed to two factors:1) The preprocessing pipeline for vocabulary generation is slightly different to the Vec2dyanmics. 2) Embedding generation was done using BERT rather than Word2Vec.

Furthermore, the findings of research indicate that Sci-Bert was able to identify different algorithms and techniques and was able to map it to the area of machine learning. In fact, Sci-Bert was not only able to map the domain of machine learning but also for other sub-topics like "markov models", "Reinforcement learning", "Component analysis", "time series" etc. The prominent thing about "Sci-Bert" algorithm is once the embeddings are

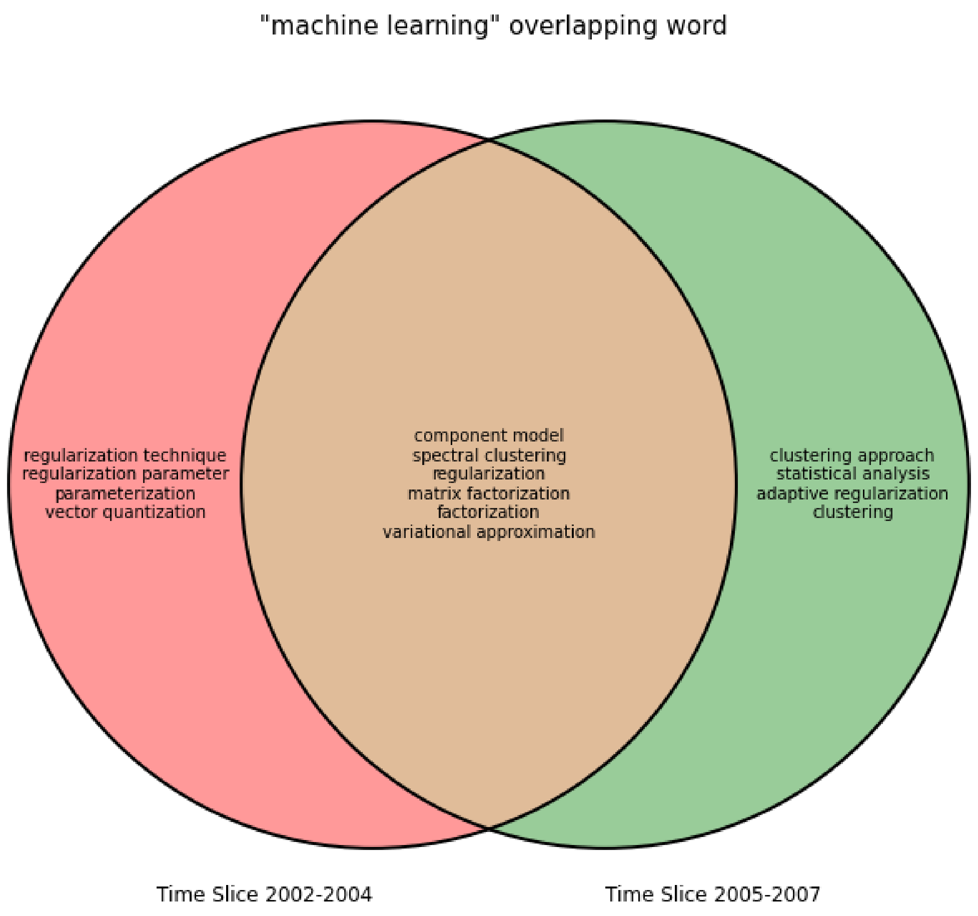


Figure 4.7: Overlapping Topics (words) for "Machine Learning" keyword for timeline 2002-2005 and 2006-2008

generated for the corpus in a temporal fashion, the history of any keyword of interest can be tracked. The code of the Sci-Bert is available at (https://github.com/xahram/Sci-Bert/blob/main/Sci_Bert.ipynb) and appendix A.1.

Overall, the findings of the "Sci-Bert" algorithm show immense potential in tracking the history of machine learning to great extent. That can be evidenced by the machine learning timeline discussed earlier. In fact, "Sci-Bert" is immensely detailed in terms of showing the topics and techniques studied by the researchers in the machine learning domain over the years. The numerical and visual approaches both justify this claim by drawing both the pictorial evidence and figures to illustrate this case.

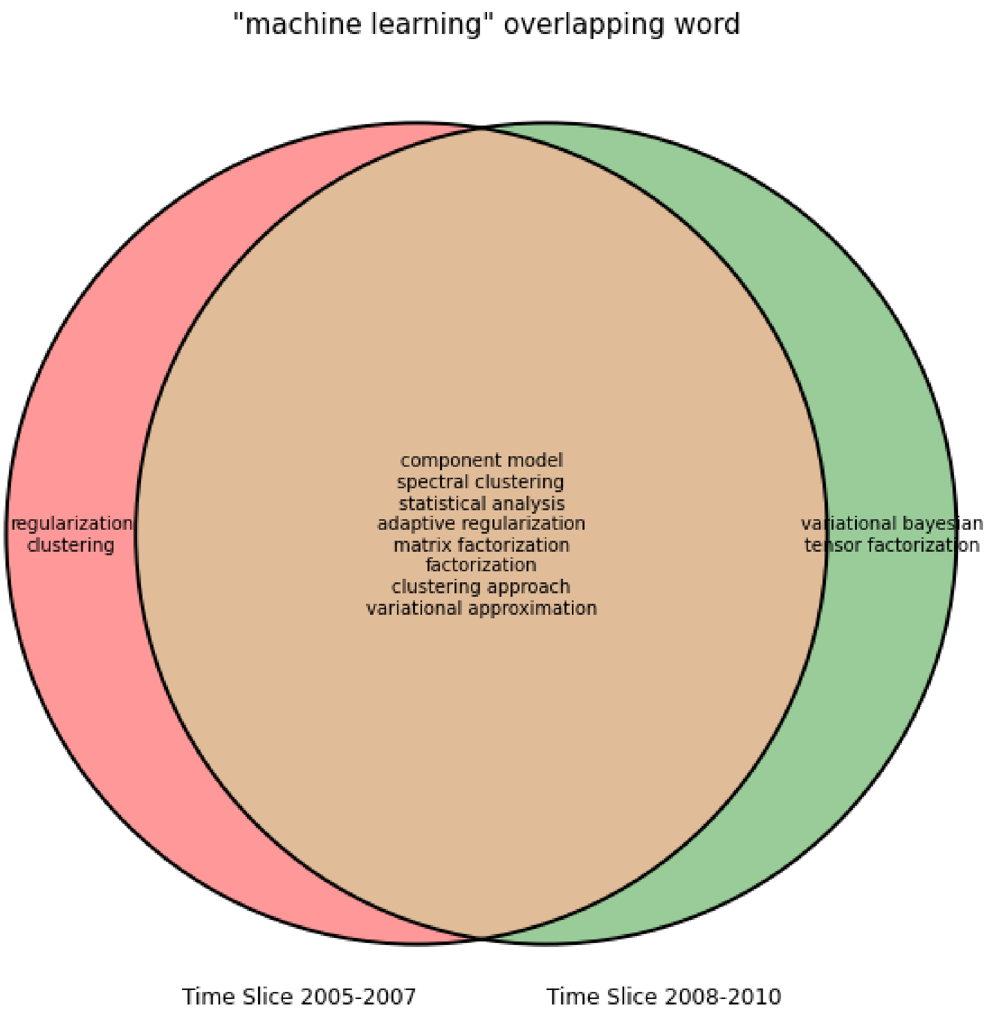


Figure 4.8: Overlapping Topics (words) for "Machine Learning" keyword for timeline 2006-2008 and 2008-2010

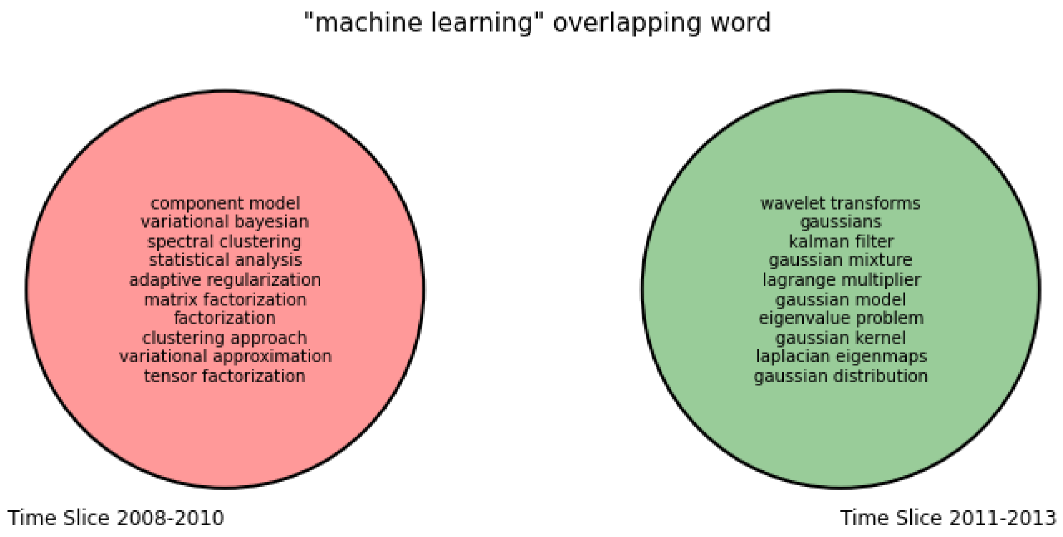


Figure 4.9: Overlapping Topics (words) for "Machine Learning" keyword for timeline 2008-2010 and 2011-2013

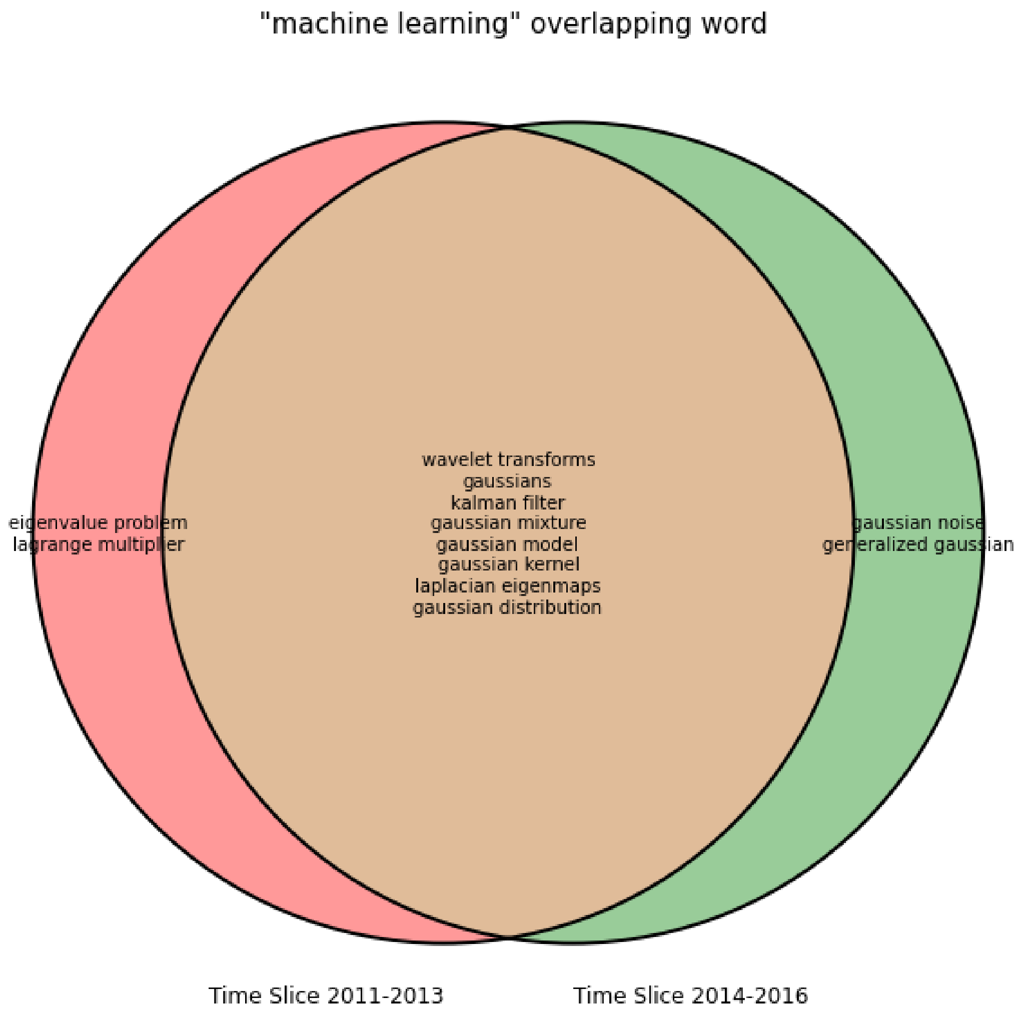


Figure 4.10: Overlapping Topics (words) for "Machine Learning" keyword for timeline 2011-2013 and 2014-2016

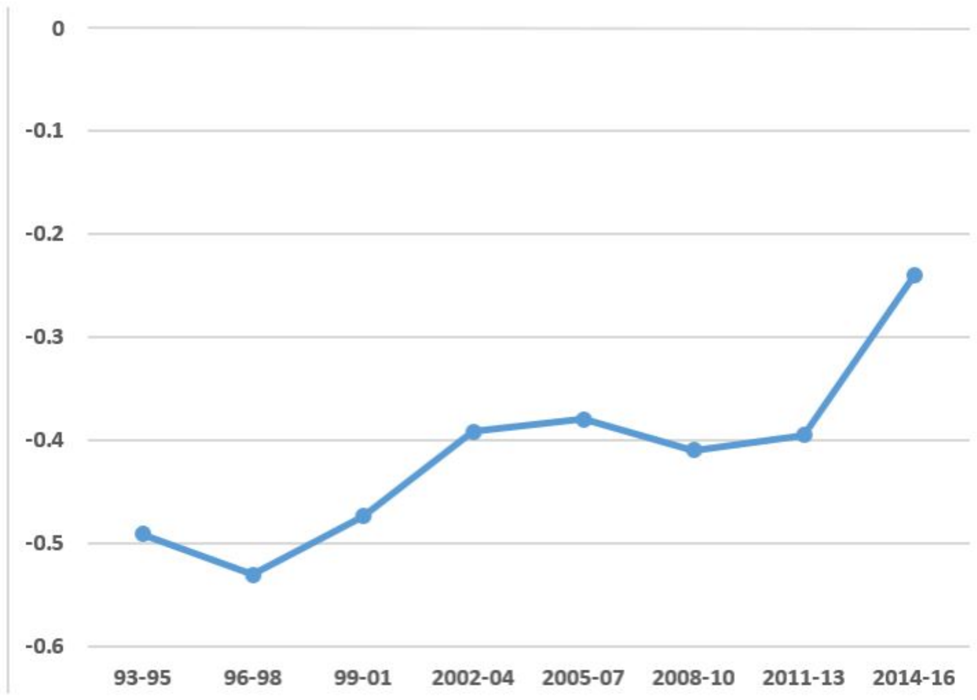


Figure 4.11: Stability measure of K-NN of Vec2dynamics Dridi et al. (2022)

# Chapter 5

# Conclusion

## 5.1 Conclusions

The research aims to investigate the dynamism of keywords in scientific literature, mainly in the domain of Machine Learning. The research is an extension of Dridi et al. (2022) and hence, follows a similar approach to draw comparative analysis.

In this study, we take a look at the significance and relevance of investigating dynamism in scientific literature. Literature review provides a brief overview of the efforts to map the linguistic evolution and dynamism in academia. The literature review highlights the research gap in the implemented methodologies. The gap provides the basis of our research aim, objectives and hypothesis. The methodology section, gives detailed explanation and rationale for the proposed experimental setup. The methodology follows acquisition of the dataset from NeurIPS, data preprocessing, generating BERT based word embedding, applying cosine similarity to compute K-NN of keywords, evaluate changing neighbours using stability measure.

The results of the study suggest BERT being able to identify the trends and dynamism in the area of machine learning. Obtained results point out BERT based embeddings to be more effective in tracing out the techniques studied during each time window. The effectivity of BERT being able to trace the timeline of machine learning can be attributed to the contextual embedding it generates. Here it's worth mentioning that the way BERT embeddings were generated also significantly improve the results obtained in this report. We take every sentence that word appears in as its context and generate embeddings based on that. This can explain the BERT efficacy in mapping the techniques to the domain of machine learning. Lastly, a comparative analysis with the work done by Dridi et al. (2022) showcases similar results for the stability measure. The two differing parts were the difference in stability in the 2nd last time window and the range of stability values. The first difference can be attributed to the way BERT generates word embeddings based on the whole context of the sentence, hence generating more detailed mapping of techniques.

The second difference also stems from the same reason and the different approach applied for the vocabulary generation.

Overall, both techniques were able to trace the timeline of machine learning effectively. In case of Sci-Bert, the dynamic nature of its implementation can take further area of studies into account that can be beneficial for the future plans of the project. More on that in section 5.2.

The project adheres to the ethical guidelines and principles laid out by Birmingham City University's Ethics code, available at https://icity.bcu.ac.uk/cebe/Research/CEBE-Faculty-Academic-Ethics-Committee, https://www.bcu.ac.uk/research/areas/research-integrity/research-ethics. The ethics code was followed diligently to ensure the credibility and timely deliverable of this research.

## 5.2 Future Work

Like most projects, this research was also time bound, The study was limited by the amount of computational resource that was required to generate word embeddings. Because we were taking every single sentence for each keyword into account. The process of generating embeddings was constrained by the project timeline, as discussed in the proposal of this research.

For the future, we continue to improve upon the obtained results. The future goal is to employ both the pretrained model & fine-tuned model and draw a comparative analysis on the efficiency of both approaches in tracing the dynamism of keywords & domains in temporal fashion.

Additionally, for the future roadmap, it would be interesting to explore other possibilities, like exploring more distance measures like Jaccard similarity, Manhattan distance and Minkowski distance. Also, the effects of other machine learning models like: SVM, Chameleon Algorithm and DBScan are also under consideration. The research follows the qualitative approach. Currently, there are no clear benchmarks to evaluate the obtained results. The most important objective for future roadmap is to develop a quantitative approach that could evaluate the significance of obtained results. Lastly, the study of changing dynamics in other scientific domains other than "Machine Learning" would also be an interesting take and is currently under consideration.

# Bibliography

Abiodun, O.I., Jantan, A., Omolara, A.E., Dada, K.V., Mohamed, N.A. and Arshad, H., 2018. State-of-the-art in artificial neural network applications: A survey. Heliyon, 4(11), p.e00938. Available from: http://doi.org/https://doi.org/10.1016/j.heliyon.2018.e00938.

Anderson, A., Jurafsky, D. and McFarland, D.A., 2012. Towards a computational history of the ACL: 1980-2008. Proceedings of the ACL-2012 special workshop on rediscovering 50 years of discoveries. Jeju Island, Korea: Association for Computational Linguistics, pp.13–21. Available from: https://aclanthology.org/W12-3202.

Beltagy, I., Lo, K. and Cohan, A., 2019. SciBERT: A pretrained language model for scientific text. Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (emnlp-ijcnlp). Hong Kong, China: Association for Computational Linguistics, pp.3615–3620. Available from: http://doi.org/10.18653/v1/D19-1371.

Breiman, L., 2001. Random forests. Machine learning, 45(1), pp.5–32.

Breunig, M.M., Kriegel, H.P., Ng, R.T. and Sander, J., 2000. Lof: Identifying density-based local outliers. Proceedings of the 2000 acm sigmod international conference on management of data. New York, NY, USA: Association for Computing Machinery, SIGMOD '00, p.93–104. Available from: http://doi.org/10.1145/342009.335388.

Conway, C.M. and Christiansen, M.H., 2001. Sequential learning in non-human primates. Trends in Cognitive Sciences, 5(12), pp.539–546. Available from: http://doi.org/https://doi.org/10.1016/S1364-6613(00)01800-3.

Cortes, C. and Vapnik, V., 1995. Support-vector networks. Machine learning, 20(3), pp.273–297.

Dean, J. and Ghemawat, S., 2004. Mapreduce: Simplified data processing on large clusters. Osdi'04: Sixth symposium on operating system design and implementation. San Francisco, CA, pp.137–150.

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. Available from: http://doi.org/10.48550/ARXIV.1810.04805.

Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers). Minneapolis, Minnesota: Association for Computational Linguistics, pp.4171–4186. Available from: http://doi.org/10.18653/v1/N19-1423.

Dongare, A., Kharde, R.R. and Kachare, A.D., 2012. Introduction to artificial neural network.

Dridi, A., Gaber, M.M., Azad, R.M.A. and Bhogal, J., 2022. Vec2dynamics: A temporal word embedding approach to exploring the dynamics of scientific keywordsmdash;machine learning as a case study. Big Data and Cognitive Computing, 6(1). Available from: http://doi.org/10.3390/bdcc6010021.

Fiala, D. and Tutoky, G., 2017. Computer science papers in web of science: A bibliometric analysis. Publications, 5(4). Available from: http://doi.org/10.3390/publications5040023.

Frermann, L. and Lapata, M., 2016. A Bayesian Model of Diachronic Meaning Change. Transactions of the Association for Computational Linguistics, 4, pp.31–45. https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00081/1567370/tacl_a_00081.pdf, Available from: http://doi.org/10.1162/tacl_a_00081.

Friedman, J., Hastie, T. and Tibshirani, R., 2000. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). The annals of statistics, 28(2), pp.337–407.

Gao, Q., Huang, X., Dong, K., Liang, Z. and Wu, J., 2022. Semantic-enhanced topic evolution analysis: a combination of the dynamic topic model and word2vec. Scientometrics, 127. Available from: http://doi.org/10.1007/s11192-022-04275-z.

Gupta, V., Sachdeva, S. and Dohare, N., 2021. Chapter 8 - deep similarity learning for disease prediction. In: V. Piuri, S. Raj, A. Genovese and R. Srivastava, eds. Trends in deep learning methodologies, Academic Press, Hybrid Computational Intelligence for Pattern Analysis, pp.183–206. Available from: http://doi.org/https://doi.org/10.1016/B978-0-12-822226-3.00008-8.

Hall, D., Jurafsky, D. and Manning, C.D., 2008. Studying the history of ideas using topic models. Proceedings of the 2008 conference on empirical methods in natural language processing. Honolulu, Hawaii: Association for Computational Linguistics, pp.363–371. Available from: https://aclanthology.org/D08-1038.

Hamilton, W.L., Leskovec, J. and Jurafsky, D., 2016. Diachronic word embeddings reveal statistical laws of semantic change. Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers). Berlin, Germany: Association for Computational Linguistics, pp.1489–1501. Available from: http://doi.org/10.18653/v1/P16-1141.

Han, J., Kamber, M. and Pei, J., 2012. 2 - getting to know your data. In: J. Han, M. Kamber and J. Pei, eds. Data mining (third edition), Boston: Morgan Kaufmann, The Morgan Kaufmann Series in Data Management Systems, pp.39–82. Third edition ed. Available from: http://doi.org/https://doi.org/10.1016/B978-0-12-381479-1.00002-2.

Han, X., 2020a. Evolution of research topics in lis between 1996 and 2019: an analysis based on latent dirichlet allocation topic model. Scientometrics, 125(3), pp.2561–2595. Available from: http://doi.org/10.1007/s11192-020-03721-0.

Han, X., 2020b. Evolution of research topics in LIS between 1996 and 2019: an analysis based on latent Dirichlet allocation topic model. Scientometrics, 125(3), pp.2561–2595. Available from: http://doi.org/10.1007/s11192-020-03721-.

Hochreiter, S., 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 06(02), pp.107–116. https://doi.org/10.1142/S0218488598000094, Available from: http://doi.org/10.1142/S0218488598000094.

Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural Comput., 9(8), p.1735–1780. Available from: http://doi.org/10.1162/neco.1997.9.8.1735.

Järvelin, A., Järvelin, A. and Järvelin, K., 2007. s-grams: Defining generalized n-grams for information retrieval. Information Processing  Management, 43(4), pp.1005–1019. Available from: http://doi.org/https://doi.org/10.1016/j.ipm.2006.09.016.

Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: F. Pereira, C. Burges, L. Bottou and K. Weinberger, eds. Advances in neural information processing systems. Curran Associates, Inc., vol. 25. Available from: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Kulkarni, V., Al-Rfou, R., Perozzi, B. and Skiena, S., 2014. Statistically significant detection of linguistic change. Available from: http://doi.org/10.48550/ARXIV.1411.3315.

LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. nature, 521(7553), pp.436–444.

Malaterre, C., Lareau, F., Pulizzotto, D. and St-Onge, J., 2021. Eight journals over eight decades: a computational topic-modeling approach to contemporary philosophy of science. Synthese, 199(1), pp.2883–2923. Available from: http://doi.org/10.1007/s11229-020-02915-6.

Martinc, M., Kralj Novak, P. and Pollak, S., 2020. EnglishLeveraging contextual embeddings for detecting diachronic semantic shift. Proceedings of the 12th language resources and evaluation conference. Marseille, France: European Language Resources Association, pp.4811–4819. Available from: https://aclanthology.org/2020.lrec-1.592.

Marín-Marín, J.A., López-Belmonte, J., Fernández-Campoy, J.M. and Romero-Rodríguez, J.M., 2019. Big data in education. a bibliometric review. Social Sciences, 8(8). Available from: http://doi.org/10.3390/socsci8080223.

Merchant, A., Rahimtoroghi, E., Pavlick, E. and Tenney, I., 2020. What happens to bert embeddings during fine-tuning? Available from: http://doi.org/10.48550/ARXIV.2004.14448.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality. Available from: http://doi.org/10.48550/ARXIV.1310.4546.

Mosbach, M., Andriushchenko, M. and Klakow, D., 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. Available from: http://doi.org/10.48550/ARXIV.2006.04884.

Pesta, B., Fuerst, J. and Kirkegaard, E.O.W., 2018. Bibliometric keyword analysis across seventeen years (2000–2016) of intelligence articles. Journal of Intelligence, 6(4). Available from: http://doi.org/10.3390/jintelligence6040046.

Popescu, O. and Strapparava, C., 2015. Semeval 2015, task 7: Diachronic text evaluation. pp.870–878. Available from: http://doi.org/10.18653/v1/S15-2147.

Reddy, N., Singh, P. and Srivastava, M.M., 2020. Does bert understand sentiment? leveraging comparisons between contextual and non-contextual embeddings to improve aspect-based sentiment models. arXiv preprint arXiv:2011.11673.

Taigman, Y., Yang, M., Ranzato, M. and Wolf, L., 2014. Deepface: Closing the gap to human-level performance in face verification. Proceedings of the ieee conference on computer vision and pattern recognition (cvpr).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., 2017. Attention is all you need. Available from: http://doi.org/10.48550/ARXIV.1706.03762.

Wevers, M. and Koolen, M., 2020. Digital begriffsgeschichte: Tracing semantic change using word embeddings. Historical Methods: A Journal of Quantitative and Interdisciplinary History, 53(4), pp.226–243. https://doi.org/10.1080/01615440.2020.1760157, Available from: http://doi.org/10.1080/01615440.2020.1760157.

Wu, Y.c. and Feng, J.w., 2018. Development and application of artificial neural network. Wireless Personal Communications, 102(2), pp.1645–1656.

Yao, Z., Sun, Y., Ding, W., Rao, N. and Xiong, H., 2018a. Dynamic word embeddings for evolving semantic discovery. New York, NY, USA: Association for Computing Machinery, WSDM '18, p.673–681. Available from: http://doi.org/10.1145/3159652.3159703.

Yao, Z., Sun, Y., Ding, W., Rao, N. and Xiong, H., 2018b. Dynamic word embeddings for evolving semantic discovery. Proceedings of the eleventh ACM international conference on web search and data mining. ACM. Available from: http://doi.org/10.1145/3159652.3159703.

# Appendix A

# Code

## A.1 Link To Sci-Bert Code

https://github.com/xahram/Sci-Bert/blob/main/Sci_Bert.ipynb

# Appendix B

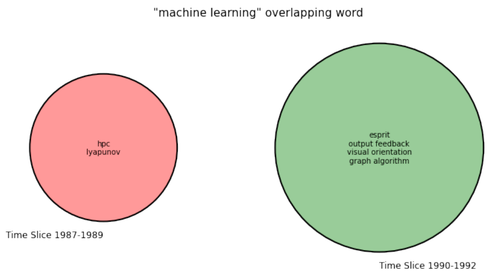
# Dot Product Similarity For Machine Learning

## B.1 Dot Product K-NN stability of Candidate Keywords

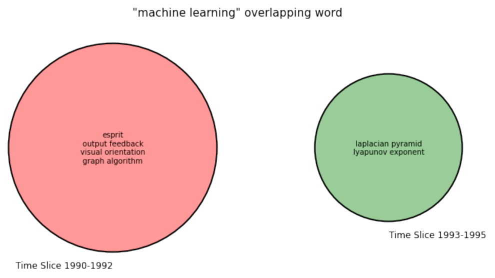
Table B.1: Dot Product K-NN Stability of top 20 Candidate keywords studied

| Keywords | 93-95 | 96-98 | 99-01 | 02-04 | 05-07 | 08-10 | 11-13 | 14-16 |
|---|---|---|---|---|---|---|---|---|
| monte carlo | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| gradient descent | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| supervised | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| reinforcement learning | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| guassian | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| graphical model | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| dynamic programming | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| neural network | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| time series | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| deep learning | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| machine learning | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| markov | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| supervised learning | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| artificial neural | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| component analysis | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| nearest neighbor | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| gradient | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| model | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| learning | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |
| Average Stability | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 | -1.0 |

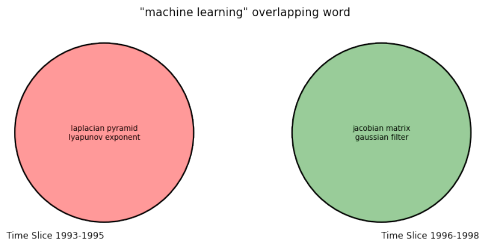
## B.2 Dot Product K-NN Venn Diagrams and Stability Measure



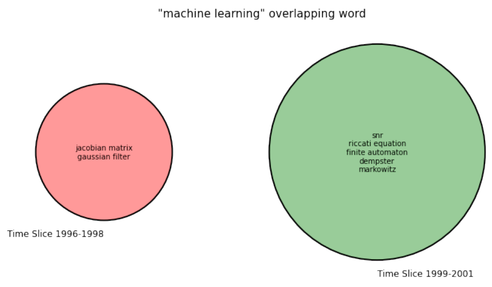
Dot Product Similarity for timeline 1987-1989 and 1990-1992



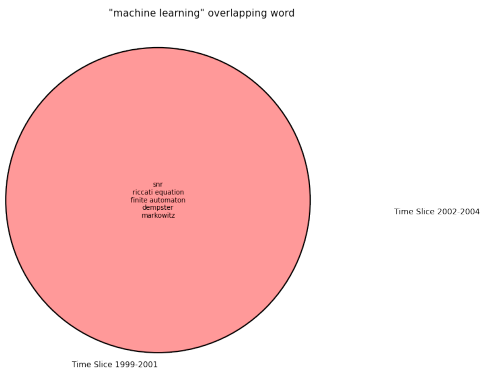
Dot Product Similarity for timeline 1990-1992 and 1993-1995



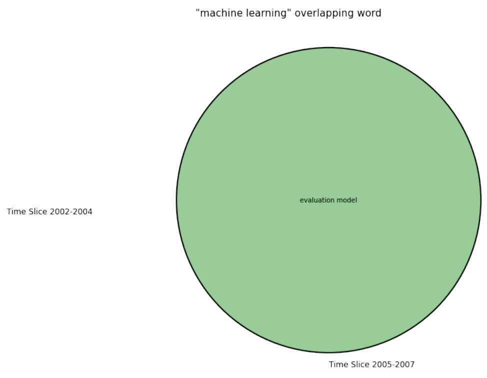
Dot Product Similarity for timeline 1993-1995 and 1996-1998



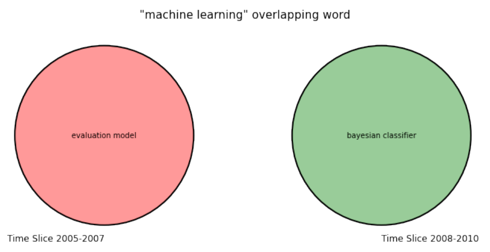
Dot Product Similarity for timeline 1996-1998 and 1999-2001



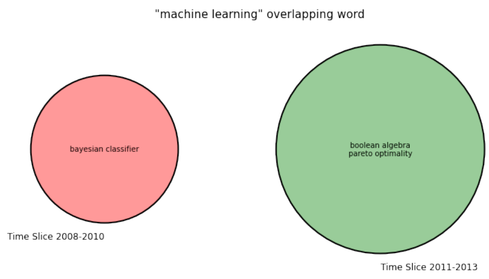
Dot Product Similarity for timeline 1999-2001 and 2002-2004



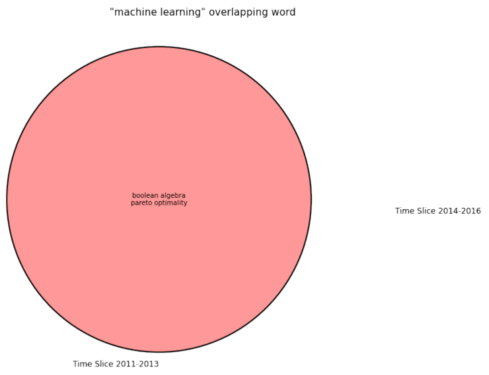
Dot Product Similarity for timeline 2002-2004 and 2005-2007



Dot Product Similarity for timeline 2005-2007 and 2008-2010



Dot Product Similarity for timeline 2008-2010 and 2011-2013



Dot Product Similarity for timeline 2011-2013 and 2014-2016

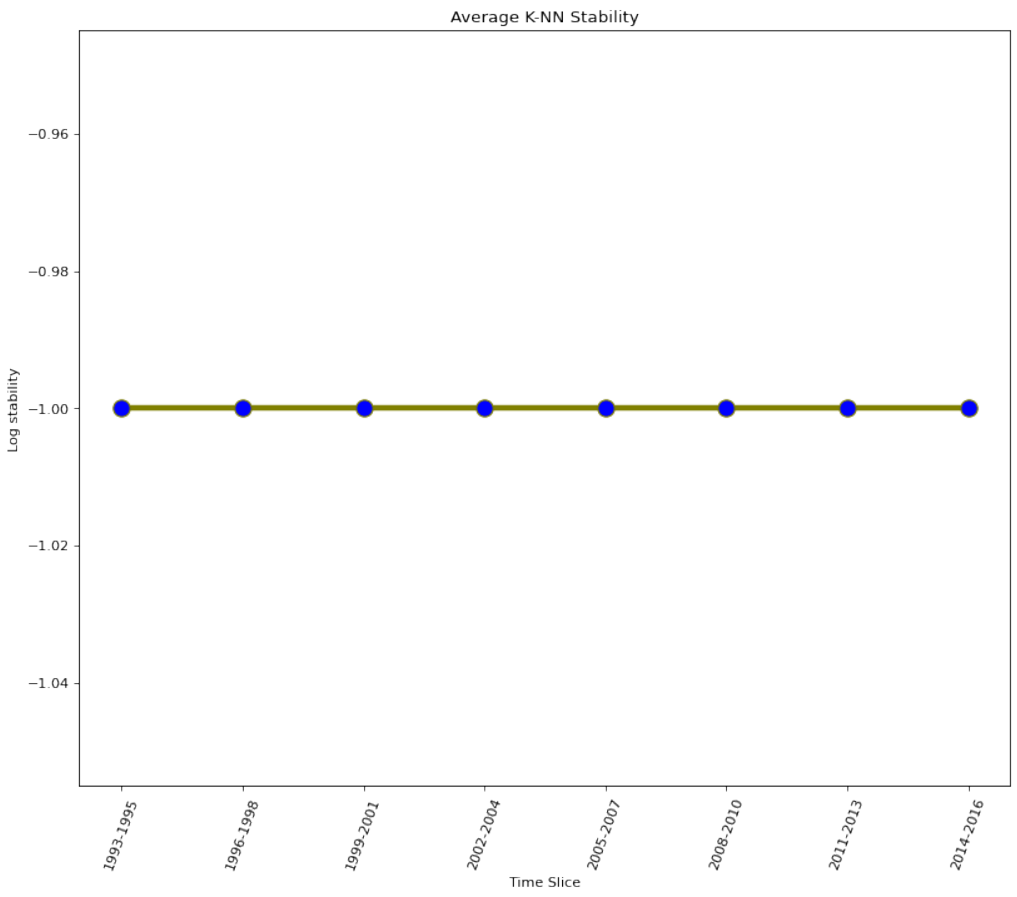


Figure B.1: Dot Product K-NN Stability from 1987 to 2016