Towards New Material Discovery from CdTe Solar Cell Literature with Machine Learning

A thesis submitted to Middlesex University in partial fulfilment of the requirements for the degree of MSc by research

Xiaolei Liu

M00732982

Department of Computer Science

Faculty of Science and Technology

Middlesex University London

April 2021

Table of Contents

Abstract

CdTe solar cells are the most successful second-generation solar technology and produce the lowest-cost electricity in the solar industry. The overarching aim of this project is to apply natural language processing (NLP) technologies to accelerate research in the field of CdTe photovoltaic devices by automatically discovering new material applications. The NLP technologies use various language models to extract most similar words. Consequently, a knowledge diagram is established by connecting these relevant similar words. The Language models include word2vec, GloVe, fastText and BERT, which are trained on a dataset of more than 22,500 paper abstracts. The performance of these language models is evaluated using a custom test dataset. The test dataset consists of 62-word pairs, which are conceptually related in the field of CdTe solar cells. The more similar the first word is to the second word in a word pair, the higher the trained language model scores. The goal of evaluating the trained language model is to find the related concepts in more similar words. The GloVe model achieves the highest score with the custom test dataset. The knowledge diagram established in this work shows the relationships between materials and concepts of interest. In addition, the language model trained on consecutive periods is used to track the timeline of material applications. The top 500 most similar words to “defect” are tracked with timeline and “selenium” is observed to appear in the GloVe model trained on paper abstracts between 2010 and 2020. This corresponds to a journal paper abstract published in 2019, which discussed the selenium passivation effect on the bulk defects of CdTe. Therefore, the knowledge diagram and timeline of material applications provide useful insights for future research and will accelerate material discoveries in the field of CdTe solar cells.

Acknowledgements

This thesis never would have become a reality without the encouragement and assistance of many wonderful people including my supervisors, family and friends. I am extremely grateful to all of you for your contributions.

I would like to thank my supervisors, Dr Kai Xu and Dr David Windridge, for their technical advice from a strategic perspective, and positive outlook on the tortuous path that a researcher always experiences. Your willingness to share experience and knowledge is truly an asset. I also give my deep gratitude to Professor Michael Walls at Loughborough University. Thank Michael for advising me on the development of CdTe solar cells.

.

Chapter 1 Introduction

Climate change is the most urgent global problem facing mankind and one which requires urgent action. Replacing electricity generation by burning fossil fuels with energy from renewable sources such as wind and solar is essential. The photovoltaic (PV) market could contribute £25 billion to the UK economy by 2030, supporting 50,000 full time jobs across the British solar supply chain [1]. By 2050 PV generated electricity could represent 20% of the UK energy mix, contributing greatly to the UK's legally binding CO_2 emission targets as well as energy security. The cost of solar generated electricity has reduced dramatically over the past decade and is now as low as £45/MWh in the UK and, as a result, the industry is again booming. However, much more can be achieved. Reducing costs by increasing module efficiency, will increase demand causing faster substitution of fossil fuels and lower consumer cost of electricity. Thin film solar PV cell/module accounted for about 5% of global shipments in 2020. First Solar, the world largest CdTe solar cell/module manufacturer, accounted for 4% of total shipments (i.e., about 80% of thin film shipments) [2]. As the most successful second generation solar technology, polycrystalline CdTe thin film solar cells have demonstrated > 22% device efficiency and are fully commercialised products [3]. CdTe photovoltaics produces the lowest-cost electricity in the solar industry and undercuts fossil fuel-based sources in many regions of the world [4]. Further efficiency improvement can lead to even lower costs for solar-generated electricity. However, searching ideal semiconductor compounds and optimising their properties to improve the CdTe device performance is a challenging and time-consuming process [5].

Machine learning is most suitable for processing and analysing text from million research papers and learn science from these resources. Therefore, natural language processing, one of the machine learning technologies, is proposed to uncover underlying patterns in the large text corpus. The most fundamental technique in NLP is to discover syntactic and semantic relationships between words using language models to assign high-dimensional embeddings to words. The distributed representations of words in a vector space encode linguistic patterns. These patterns can simply be represented as linear translations [6]. For example, the vector analogy calculation vec("China") – vec("Beijing") + vec("Tokyo") results in a vector closer to vec("Japan") than any other word vector. In this work, the non-contextual language models of word2vec, GloVe and fastText and the contextual language model of BERT are used to explore these relationships.

1.1 Research Objective and Motivation

The overarching aim of this research project is to apply various language models of word2vec, GloVe, fastText and BERT in NLP to discover the syntactic and semantic relationships between materials and concepts in the field of CdTe solar cells. From these relationships, a knowledge diagram is constructed by connecting the related most similar material and concepts in the vector space. Exploring the knowledge diagram provides new insights for promising new material applications and consequently improve the performance of CdTe photovoltaics. The machine learning model developed in photovoltaic materials, i.e., CdTe solar cells, can be extended to other material domains and provide a toolkit of data-driven material discovery for the community at large.

The static language model, such as word2vec, has been used to extract the relatedness of materials and concepts [7]. However, the contextual language model including BERT has not been investigated for this specific application. In this work, the word embeddings in different layers of a BERT model are applied to construct this relationship. In addition, GloVe and fastText language models are also exploited in this work and the model performance of word2vec, GloVe, fastText and BERT is compared.

1.2 Outline

Chapter 2 reviews the application of machine learning methods in the material discovery and synthesis. The principles of word2vec, GloVe, fastText and BERT algorithms are also discussed.

In Chapter 3, the language model training methods including text pre-processing and model training hyperparameters are introduced. A test dataset is set up to evaluate the performance of word2vec, GloVe, fastText and BERT models. The training hyperparameters of epoch, initial learning rate and minimum word occurrence are introduced for the non-contextual language models, i.e., word2vec, GloVe and fastText. In the training of BERT, the contextual language model, the pooling and layer choice strategy, the contextual word embedding representation with the first principal component are discussed.

The trained models are evaluated using the test dataset. The model performance is shown in Chapter 4. The effects of window size, vector dimension size and negative sampling of word2vec, GloVe and fastText models are discussed. The effect of layer choice and learning rate on the BERT model performance is demonstrated.

Chapter 5 shows the applications of the trained language model. The GloVe model is used to construct a knowledge diagram from the relatedness between materials and concepts. The material applications are tracked using the timeline when the material appears in the most similar words of a target material and concept. Finally, the conclusions and future work are summarised in Chapter 6.

.

Chapter 2 Literature Review

2.1 Introduction

Despite the rapid development of modern material science, it currently takes an average of 15 to 20 years for a successful material to transfer from laboratory to practical applications [5]. The laborious and painful slow process demonstrates the fundamental difficulties of traditional material discovery and property characterisation. Machine learning is poised to change the conventional approach and accelerate material discovery. As a subfield of artificial intelligence, machine learning explores big data to find underlying patterns and provide new insights into material discovery. The innovative application of machine learning is to process and analyse text from million research papers and learn science from these resources [8]. The recent deep-neural-network machine learning has been widely used in natural language processing, which shows a flurry of state-of-the-art results in many natural language tasks. The natural language processing techniques including word embeddings, recurrent neural network and long short-term memory (LSTM), analyse text to extract metadata from content such as entities, keyworks, relations and semantic roles. These deep neural networks have become a new paradigm for end-to-end learning of a high-level task, for example question answering and machine translation that directly learn sequence-to-sequence transformations.

Machine learning has been explored in material discovery and produced some exciting results. Ceder et al applied text mining by training a neural network on three million abstracts of journal articles published before 2005 [8]. This model successfully predicts that a photovoltaic material with suitable band gap has potential applications for thermoelectric material. The prediction is verified by later thermoelectric application of this photovoltaic compound after 2005.

Other researchers have used machine learning to analyse material synthesis conditions from journal articles and extract material recipes of new compounds [8] [9]. Olivetti et al have applied natural language processing techniques to automatically compile material synthesis parameters and trends across tens of thousands of research papers [10]. This framework is then used to correlate the synthesis conditions and resulting topology and produce incredible results. Researchers at Haverford

College digitised their lab notebooks and recorded worked and unsuccessful reaction conditions for the crystallisation of vanadium selenites. A neural network has been trained on these data and achieved 89% accuracy for predicting the success of a new set of reaction parameters [8]. Researchers previously adopted an approach of manually reading through journal papers and extracting useful information. Machine learning provides an efficient pipeline for identifying and analysing research articles in an automated fashion.

Tshitoyan el al. applied word2vec to analyse massive body of scientific literature and predict new functional materials [7]. The word2vec model creates linguistic contexts of words from a large corpus of text and produces word embeddings in several hundred dimensions. The words with similar meanings will accordingly have similar word embeddings. The authors encoded the materials science knowledge in the published literature as information-dense word embeddings without manual labelling and supervision. It has been demonstrated that materials with similar properties cluster together in the word embedding space. The similarity between materials is calculated by projecting the normalized word embeddings (dot product). The approach also supports domain-specific material analogies, such as ferromagnetic-NiFe+IrMn≈antiferromagnetic. The analogies are expressed as the nearest word to the result of subtraction and addition operations between the embeddings (Figure 2.1a). The main novelty of the word embeddings is to predict new functional materials. For example, the model predicts 7,663 compounds similar to the "thermoelectric" word embedding. These materials are not reported explicitly in the text corpus. However, density functional theory (DFT) predicts these thermoelectric compounds. The word2vec model is also tested to predict thermoelectric materials reported later in the literature from the model trained at various points in the past. The results indicate the top 50 similar word embedding-based predictions are more likely to have been investigated as thermoelectrics within the next five years. The series of connections between word embeddings is examined to lead to new material predictions. If material A is relevant to B and B relates to C, the material A will be connected to C. $CsAgGa_2Se_4$ appears next to "chalcogenide", "band gap", "optoelectronic" and "photovoltaic applications" word embeddings, which are largely overlapped by thermoelectric. The embedding correlations consequently predict $CsAgGa_2Se_4$ is a thermoelectric material (Figure 2.1b). The reported word2vec model extracts knowledge and relationship from massive scientific literature and recommends functional materials several years before their discovery. The findings pave a pathway towards mining scientific literature and discovering new materials.

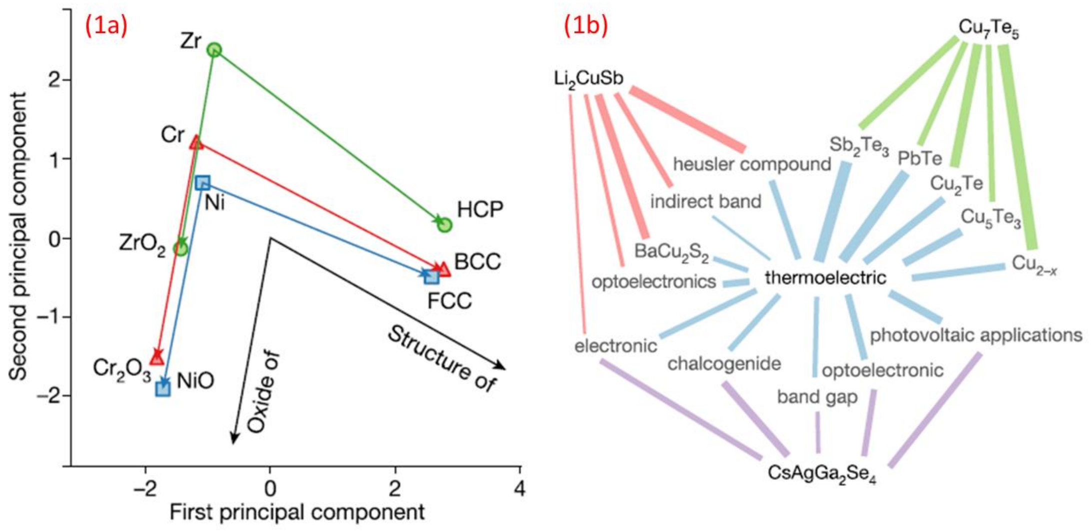

Figure 2.1 a, Principal component analysis of word embeddings for Zr, Cr and Ni with their principal oxides and crystal symmetries. The relative positioning of the word embeddings encodes material relationships, such as “oxide of” and “structure of”. b, Prediction of new thermoelectric material $CsAgGa_2Se_4$ using series of connections between materials and concepts in a knowledge diagram. Reprinted with permission from [7].

2.2 Language Models

In this work, the language models of word2vec, GloVe, fastText and BERT are trained on tens of thousands published paper abstracts. Word2vec generates distributed embeddings in a vector space, which is one of the most popular representations of a specific word in a document vocabulary. The word embeddings in a word2vec language model can grab semantic and syntactic similarity of words in the context of a document and establish a relationship between words.

As a shallow window-based method, word2vec relies on the local window context and fails to exploit the large amount of repletion in the data corpus. The window-based method suffers from the disadvantage that it does not use the global co-occurrence statistics of the corpus. On the other hand, Global Vectors for Word Representation

(GloVe) combines the advantages of global matrix factorisation and local context window approaches [11]. Unlike word2vec, GloVe uses not only local statistics (i.e. local context windows) but also global statistics (i.e. word co-occurrence matrix) to generate word vectors.

The word2vec word embedding ignores the internal structures of words and generates a unique word vector for every individual word. This method becomes a significant limitation for morphologically rich languages, such as German or Finnish. These morphologically rich languages include many word forms that occur rarely (or not at all) in the training corpus, making it difficult to learn good word representations [12]. FastText addresses these limitations by representing each word as an aggregation of its subwords, i.e., character n-grams. FastText extends the continuous skip-gram model in word2vec by considering morphology of words and representing a rare word in a large vocabulary as the sum of its character n-gram vectors.

Word2vec represents a word with a single word vector regardless of context. Therefore, these word embeddings are static. On the other hand, BERT (Bidirectional Encoder Representations from Transformers) creates contextualised word embeddings, which are sensitive to the context incorporated from both directions. For example, given two sentences [13]: “The man was accused of robbing a bank.” “The man went fishing by the bank of the river.” Word2vec would create the same word vector for the word “bank” in both sentences. However, BERT produces different word embeddings for the word “bank” regarding its context. These context-informed word embeddings represent more accurate features of the words, leading to better model performance in the down-stream tasks.

2.2.1 Word2vec

There is a long history to represent words as continuous vectors. Some popular language models, such as neural network language model (NNLM) [14] and recurrent neural net language model (RNNLM) [15], were previously proposed. However, a non-linear hidden layer in these models adds computational complexity. Therefore, word2vec was proposed to address this complexity by removing the non-linear hidden layer and sharing the projection layer for all words [6]. The most significant feature of the word2vec model is that the words with similar context will occupy close spatial positions. The word similarity is mathematically measured using the cosine of the angle between the word embeddings.

There are two approaches in a word2vec model to construct word embeddings: Continuous Bag-of-Words (CBOW) Model and Continuous Skip-gram Model (Figure 2.2).

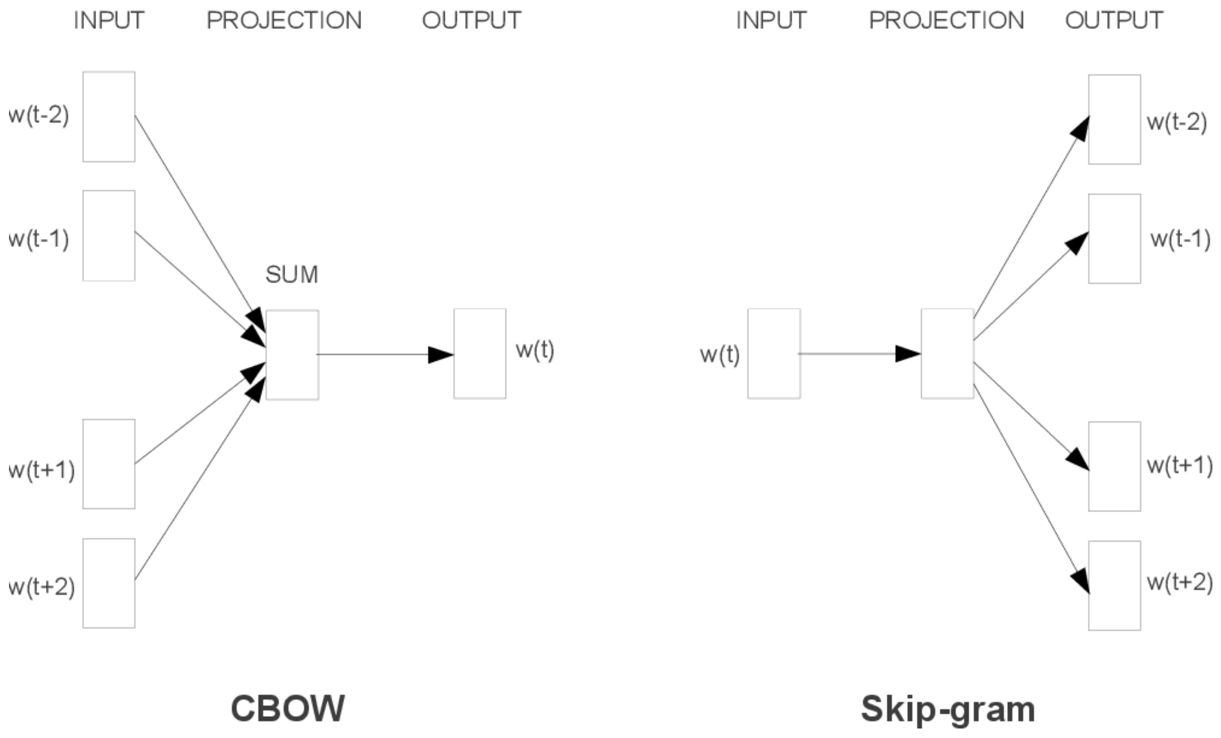

Figure 2.2 CBOW and Skip-gram model architectures. Reprinted with permission from [16].

The CBOW model uses the context words as the input and predict the target word corresponding to the context. The Skip-gram model uses the target word as the input and outputs the context words [6]. The hidden layer neurons in the CBOW and Skip-gram models do not have activation functions (e.g., sigmoid, tanh and ReLU) and just project the weighted sum of inputs. The only non-linearity of these models is the softmax function in the output layer.

Stochastic gradient descent and back-propagation are applied to train the word2vec language model. However, learning the output vectors is expensive without computing optimisation. To avoid over-fitting, a large amount of training data is needed to tune many weights. For each training sample, a great number of weights in the weight matrix are needed to be updated. Therefore, these weights times the training data result in a very slow training process. To improve the computing efficiency, hierarchical softmax and negative sampling are proposed to limit the number of output vectors that must be updated per training sample [6]. Hierarchical softmax reduces the computational complexity from O(V) to O(log(V)) per training instance per context word [17]. Negative sampling only updates the weights associated with a small number of randomly selected “negative” words and the “positive” word. Therefore, this significantly reduces the computing cost.

2.2.2 GloVe

The GloVe model has a fasted training speed than word2vec. It optimises one count at a time. However, word2vec optimise one window at a time. The window-based approach might train over the same co-occurrence multiple times. The global word co-occurrence matrix provides a primary source of information for learning word vectors. The efficient use of global statistics make the GloVe model perform well even on small corpus and small vector sizes.

Let $P_{ij} = P(j/i)$ be the probability that word j appears in the context of word i. Table 3.1 tabulates the co-occurrence probabilities and ratio of these probabilities for target words ice and steam with selected context words from a 6 billion token corpus [11].

Table 2.1 Co-occurrence probabilities for target words ice and steam with selected context words. Reprinted with permission from [11].

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

The ratio of the two probabilities is better able to derive the semantic relationships between words in contrast to the raw probability. For example, taking the two target words i=ice and j=steam, the ratio of the two co-occurrence probabilities is either large or small for the context words of solid and gas because these context words are relevant to either of the two target words. However, for the context words of water and fashion, the ratio is close to one because these two context words are either related to both the two target words, or to neither.

Leveraging a new weighted least squares regression model and a weighting function $f(X_{ij})$, the cost function is [11]

$$J = \sum_{i,j=1}^{V} f(X_{ij})\left(w_i^T w_j + b_i + b_j - logX_{ij}\right)^2$$

Where V is the size of the vocabulary, w_i and w_j are word vectors of the target and context word, X_{ij} is the number of times word j occurs in the context of word i. The weighting function $f(X_{ij})$ should be non-decreasing for the rare co-occurrences and relatively small for the frequent co-occurrences. Therefore, the rare co-occurrences and frequent co-occurrences are not overweighted. The GloVe model uses one class of functions as the weighting function (Figure 2.3),

$$f(x) = \begin{cases} (x/x_{max})^{\alpha}, & x < x_{max} \\ 1, & otherwise \end{cases}$$

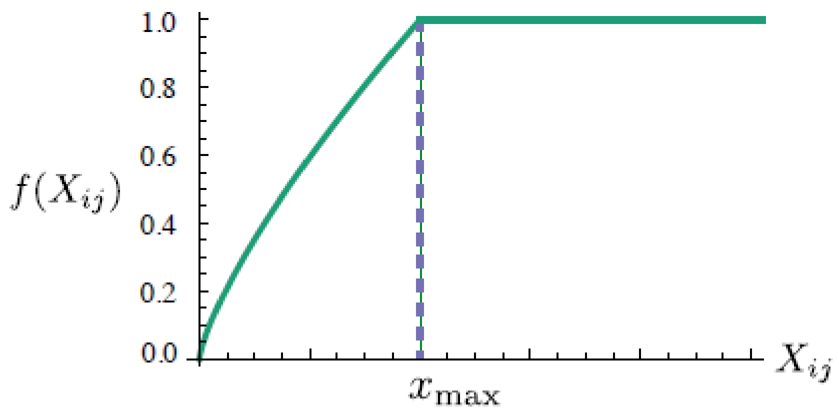

Figure 2.3 Weighting function *f(x)* with *a* = ¾. Reprinted with permission from [11].

2.2.3 fastText

FastText represents each word as a bag of character n-grams. For example, taking the word “*where”* and $n - grams = 3$, the word “*where”* will be represented by the character n-grams: <wh, whe, her, ere, re> and the special sequence <where> [12]. The special boundary symbols < and > indicate the beginning and end of words.

A skipgram model aims to maximise the following log-likelihood [12]:

$$\sum_{t=1}^{T} \sum_{c \in C_t} logp(w_c|w_t)$$

Where $p(w_c|w_t)$ is the probability of a context word w_c occurring given a target word w_t. C_t is the set of words surrounding the word w_t.

Considering negative sampling, the log-likelihood function becomes [12]:

$$\sum_{t=1}^{T}\left[\sum_{c\in C_t}\ell(s(w_t,w_c)) + \sum_{n\in\mathcal{N}_{t,c}}\ell(-s(w_t,n))\right]$$

Where $\ell(x)$ denotes the logistic loss function $log(1+e^{-x})$. $S(x)$ is the scoring function. The score is calculated as the scalar product between target and context word vectors. In the fastText model, a word is represented by the sum of the vectors of its n-grams. Therefore, the scoring function is obtained as follows [12]:

$$s(w,c) = \sum_{g\in\mathcal{G}_w} z_g^T V_c$$

Where $\mathcal{G}_w$ is the set of n-grams appearing in the word *w*. Z_g is the vector of a n-gram *g*. V_c is the word embedding of the context word *c*.

The optimal length of n-grams should cover a wide range of word information. Although the optimal choice of n-gram length depends on specific tasks and languages, the length ranges from 3 to 6 provide a reasonable amount of subword information in practice [12].

2.2.4 BERT

Transformer

BERT uses a Transformer model architecture, which relies entirely on a self-attention mechanism to build a relationship between input and output. This Transformer model architecture significantly improves computational efficiency through parallelisation. In contrast to its counterpart, the recurrent neural network factors computation along the symbol positions of input and output sequences. Therefore, this inherently sequential nature precludes parallelisation within training samples and deteriorates computational performance.

BERT have an encoder-decoder structure, using stacked self-attention and pointwise, fully connected layers (Figure 2.4). The encoder consists of a stack of 6 identical layers [18]. Each layer has two sub-layers, which apply a multi-head self-attention mechanism and a simple, position-wise fully connected feed-forward network. The decoder is also composed of a stack of 6 identical layers. Each decoder layer adds an additional multi-head attention sub-layer over the output of the encoder stack. The encoder maps an input sequence to a sequence of continuous representations. The decoder then outputs a sequence of symbols one element at a time.

Attention

The Attention mechanism learns the mapping between different parts of the input sequence and corresponding parts of the output sequence [18]. While predicting an output element, it will pay more attention to a corresponding element in the input sequence. Therefore, the Attention mechanism utilises the intermediate encoder states to construct context vectors, which the decoder uses to produce the output sequence. The self-Attention mechanism exploits the interaction between the elements in the input sequence ("self") and decides which element comes into more focus. The output of the self-Attention function accumulates these interaction and attention scores.

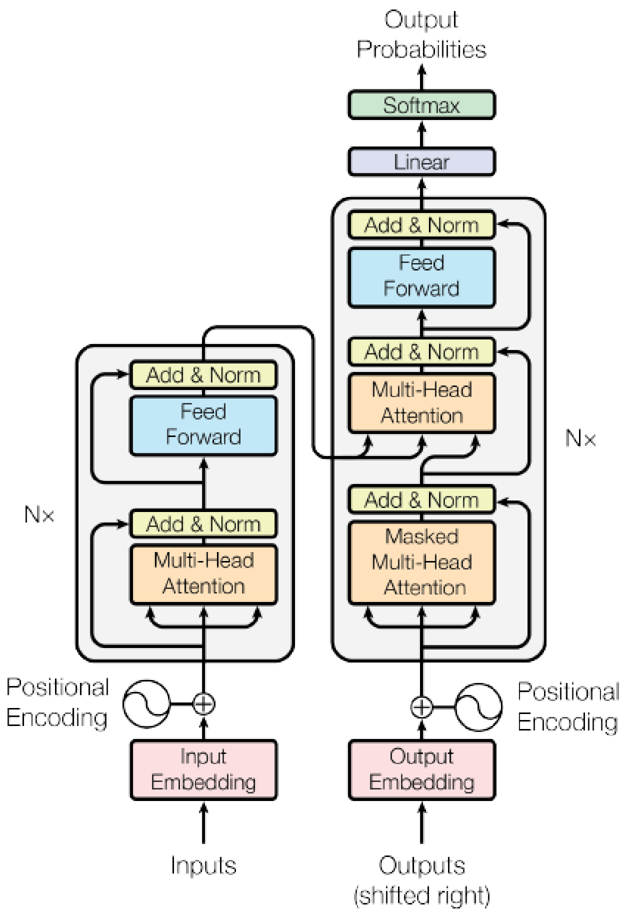

Figure 2.4 The transformer model architecture. Reprinted with permission from [18].

The Attention function maps a set of vectors, i.e., query, key, value to an output. The Attention output is calculated as [18]:

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

Q, K and V are the matrices of query, key and value vectors. $\frac{1}{\sqrt{d_k}}$ is the scaling factor, where d_k is the dimension of the key vector.

Multi-headed Attention mechanism continues to improve the performance of the Attention layer. The queries, keys and values are linearly projected h times with different, learned linear projections to d_q, d_k and d_v dimensions, respectively. It provides the attention layer multiple “representation subspaces” and gives the model’s ability to attend to different positions in the input sequence.

2.3 Summary

Various language models have been reviewed. Word2vec uses the local context to generate word embeddings. It has been exploited to extract relationship among materials from scientific literature. However, the word2vec model has some limitations. To address these disadvantages, other non-contextual language models, such as GloVe and fastText, and the contextual language model of BERT will be applied, and their performance will be compared in this work. The GloVe language model uses both local context windows and word co-occurrence matrix to generate word vectors. The fastText model considers morphology of words and represents a word as the sum of its character n-gram vectors. Therefore, it can obtain vectors even for out-of-vocabulary words. The contextual language model of BERT produces contextualised word embeddings given its context. These language models will be trained and evaluated using a customised test dataset in this work. The best performing language model will be used to construct a knowledge diagram in the vector space and track material application timeline.

Chapter 3 Language Model Training

3.1 Dataset and Text Pre-processing

The language models are trained on more than 22,500 paper abstracts about the CdTe material. These papers are published between 1909 and Jan 2020 and collected from the Scopus database. Scopus maintains a database of abstracts and citations from approximately +11,000 publishers. The keyword “CdTe” is used to retrieve these abstracts, which are in the excel format with bibliographic metadata. These abstracts summarise the discoveries in the scientific journal articles and reduce the data volume in later stages of data processing.

The text pre-processing framework in this work consists of tokenisation and normalisation [19]. Tokenisation splits longer stings into tokens, exclusively in words. The white space is used as a segmentation boundary of the word tokens. Normalisation process includes converting all text to lower case, removing punctuation, removing stop words and lemmatisation. Stop words are the most common words in a language, such as “the”, “and” and “a”. They do not greatly contribute to the meaning of a text. Therefore, the stop words are generally removed before further processing. Lemmatisation captures canonical forms based on a word’s lemma. For example, lemmatisation of “better” returns “good”. A programming snippet of the text pre-processing is shown in Appendix A.1. Python is used as the programming language in this work. NLTK provides industrial-leading NLP libraries for processing human language data. The libraries for tokenisation, stop words and lemmatisation are utilised in the text pre-processing.

3.2 Evaluation dataset

An evaluation dataset is constructed with 62-word pairs of materials and concepts (Appendix B.1). These materials and concepts of interest indicate the current research challenges and future prospects of CdTe photovoltaics. In this work, the trained language models are tested to examine the cosine similarity between the word pair. The similarity is scored on a scale ranging from 0 to 5 (Table 3.1). Taking as an example the word pair (first word: “Se”, second word: “passivate”), if “passivate” appears in the top 100 most similar words to “Se”, the language model scores 5. If “passivate” falls in a range of 101 to 200 of the most similar words to “Se”, the model will score 4 and so on. The more similar the two words are, the higher the score is. The total score of the 62-word pairs is finally calculated for evaluating the language model performance. The higher the total score is, the better the language model performs. The goal of training the language models is to assign the related materials and concepts of interest to much closer positions in the vector space.

Table 3.1 Similarity score of word pairs.

Similarity Range	Score
1 to 100	5
101 to 200	4
201 to 300	3
301 to 400	2
401 to 500	1
> 500	0

3.3 Non-contextual language model training – word2vec, GloVe and fastText

The word2vec, GloVe and fastText models are trained to detect the relationship between materials and concepts in the field of CdTe solar cells. The fastText model is trained in a Google Colab notebook due to a relatively large computing resource requirement of Random-Access Memory (RAM). The effects of window size, vector dimension size and negative sampling on model performance are investigated. The training hyperparameters for word2vec, GloVe and fastText models are listed in Table 3.2.

Table 3.2 Training hyperparameters of the word2vec, GloVe and fastText models.

Hyperparameter	Value
Epoch	20
Initial learning rate	0.03
Minimum word occurrence	2

3.4 Contextual language model training – BERT

3.4.1 Training BERT

Different layers in the BERT model encode different kinds of information. The word embeddings in different layers and appropriate pooling strategy are investigated to achieve the best results for this specific application.

A BERT language model can be trained using two methods: pre-training and fine-tuning [20]. The pre-training approach trains a BERT model on unlabelled data over different pre-training tasks. During fine-tuning, a BERT model is initialised with the pre-trained parameters, and all the model parameters are fine-tuned using labelled data from the downstream tasks.

In this work, the language model is fine-tuned on a pre-trained SciBERT model from the Allen Institute for Artificial Intelligence. The fine-tuning approach is relatively inexpensive in contrast to pre-training. The SciBERT model is trained on a random sample of 1.14M papers from Semantic Scholar. Leveraging the unsupervised pre-training on a large corpus of scientific publications, SciBERT improves performance on downstream scientific NLP tasks.

The SciBERT model is fine-tuned with a Masked Language Model task using the CdTe dataset in a Google Colab notebook. The WordPiece vocabulary size of the CdTe dataset is 6769. The special tokens of [CLS] and [SEP] are added at the beginning and end of the sequence. 15% of the token positions in the training dataset is randomly masked for prediction. The SciBERT model is fine-tuned for 5 epochs with a learning rate of 2e-5 and a batch size of 8.

3.4.2 First Principal Component Representation of Contextualised Word Embeddings

In a BERT language model, word embeddings are sensitive to their context. In each of the 12 layers of the SciBERT model, a word has various representations in different context. The distribution of these word vectors is anisotropic, occupying a narrow cone in the vector space [21]. The same word has non-identical word embeddings in different contexts. The cosine similarity becomes more dissimilar in the upper layer vector representation. This implied that the word embeddings in the upper layers of a BERT model are more task specific.

The word-sense representations of the same word in a layer of the BERT model is assigned in a proportion of variance rather than a finite number of vectors. The first principal component is applied to represent the contextualised vectors of a single word.

Principal Component Analysis (PCA) is used to calculate the first principal component. PCA is a dimensionality reduction technique with a goal to maximise the variance of a dataset projected onto a set of orthogonal axes (Figure 3.2) [22]. For a given matrix A, the k^{th} principal component is the eigenvector of the covariance matrix of A corresponding to the k^{th} largest eigenvalue. Singular value decomposition is a stable and precise method to calculate the eigenvector and eigenvalues of the covariance matrix.

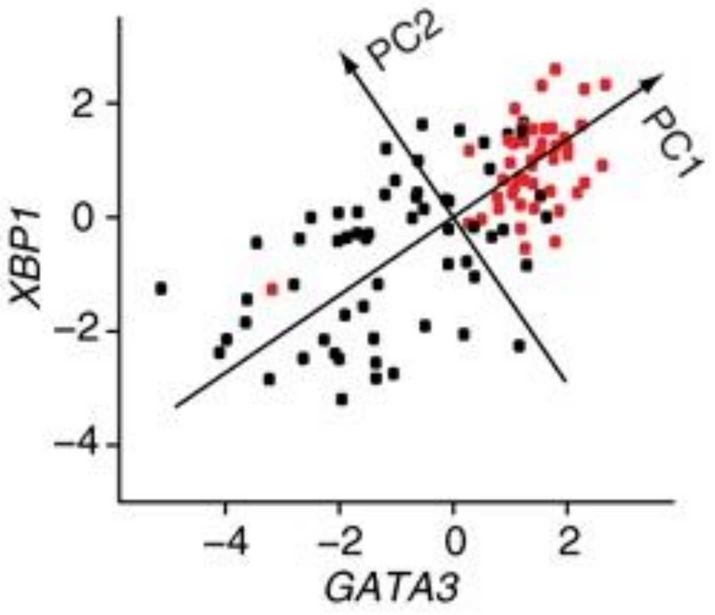

Figure 3.2 Directions of first and second principal components along which the 2D data have the largest spread. Reprinted with permission from [22].

Chapter 4 Results and Discussions

4.1 Non-contextual language models – word2vec, GloVe and fastText

The language models of word2vec, GloVe and fastText are trained on the collected paper abstracts from Scopus. The training time of word2vec, GloVe and fastText including text pre-processing is approximately 2.9, 5.26 and 4.22 minutes, respectively. These language models are optimised by adjusting the training hyperparameters of window size and word embedding size. The window size means the number of words behind and ahead of the target word. The word embedding size is the number of dimensions of a word embedding in the vector space.

The effect of window size (i.e., 2, 5, 8, 10, 15, 20 and 30) on scores of the test dataset is investigated. The word embedding size of 300 is used during the training. Figure 4.1 shows the scores of word2vec, GloVe and fastText models with different window sizes.

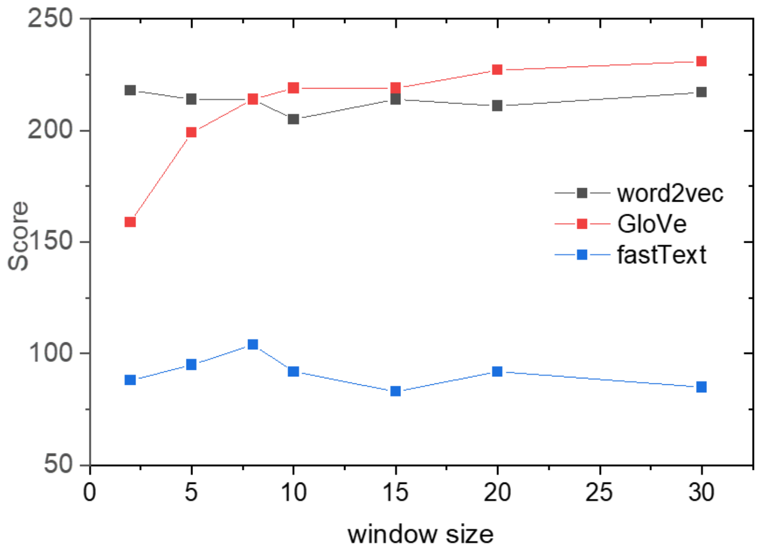

Figure 4.1 Scores of the trained word2vec, GloVe and fastText models with a window size of 5, 8, 10, 15, 20 and 30.

The window size significantly increases the scores of the GloVe language model. In general, a large window size serves better for the extraction of word relatedness, which complies with the design of the test dataset for this specific application. However, the improvement of the word2vec and fastText models is not dramatic. The word2vec and GloVe models are observed to perform much better than fastText. FastText treats each word as composed of character n-grams in the training process. The word vector consists of the sum of these character n-grams [12]. For example, where n-gram = 3, the word “aquarius” can be represented as 3-gram characters:

<aq, aqu, qua, uar, ari, riu, ius, us> and <aquarius>.

The character n-grams are used to produce embeddings for previously unseen words. Taking the word “aquarius” as an example, word2vec and GloVe models cannot create a word vector if “aquarius” did not appear in the training corpus. The reason is word2vec and GloVe models treat each word as the smallest entity to train on. However, the fastText model can produce a vector for “aquarius” by comparing the sharing character n-grams between “aquarius” and a known word “aquarium” in the vocabulary. This approach results in close word embeddings between “aquarius” and “aquarium”. Although fastText can represent Out-of-Vocabulary (OOV) words, it performs poorly for the test dataset in this specific application.

The effect of word embedding dimension size (i.e., 50, 100, 200, 300 and 600) is investigated. The window size of 30 is applied during the training of word2vec, GloVe and fastText models. These trained language models are evaluated on the test dataset for 5 times at each word embedding dimension size. The score does not change at each testing process.

Figure 4.2 shows that the performance improvement of the GloVe model diminishes for word embeddings larger than the dimension size of 200. There is an outlier at the dimension size of 50 for the word2vec language model. The fastText model exhibits slightly decreased scores with increasing vector size. The dimension size has a minor effect on the performance of the fastText model. The training of the fastText language model with a dimension size of 600 is not successful due to the limitation of Random-Access Memory (RAM). Training a fastText model consumes more computing resources than word2vec and GloVe because it uses sub-word information at each training step. A word vector in a fastText model is learned from its character n-grams and the complete word. The mean of the target word vector and its component n-grams are used in the training. At each training point, each of the component vectors that were combined to from the target word is uniformly adjusted to minimise the error. The word vectors in a fastText model contain embedded sub-word information. However, training a fastText model needs larger computing resources in contrast to word2vec and GloVe models.

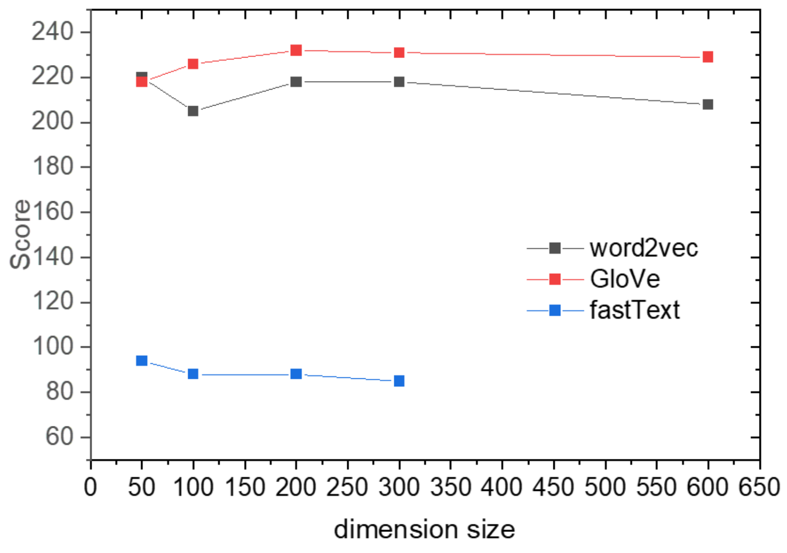

Figure 4.2 Scores of the trained word2vec, GloVe and fastText models with a vector dimension size of 50, 100, 200, 300 and 600.

Negative sampling is a critical hyperparameter in the word2vec and fastText model training. Training a neural network with gradient descent is a slow process. A huge training dataset is normally used to train a neural network to avoid over-fitting. All the neuron weights need to be slightly adjusted to predict each training sample more accurately. Therefore, millions of weights times billions of training samples means the training process is time-consuming. With negative sampling, only a small percentage of the weights ae modified, rather than all of them. Therefore, the negative sampling algorithm significantly simplifies the training process and reduces the training time [6]. Negative sampling also results in more accurate representations of frequent words. The effect of negative sampling (i.e., 2, 5, 10 and 20) is investigated for the language models of word2vec and fastText. The window size of 30 and vector dimension of 300 are used in the training process. The scores are shown in Figure 4.3.

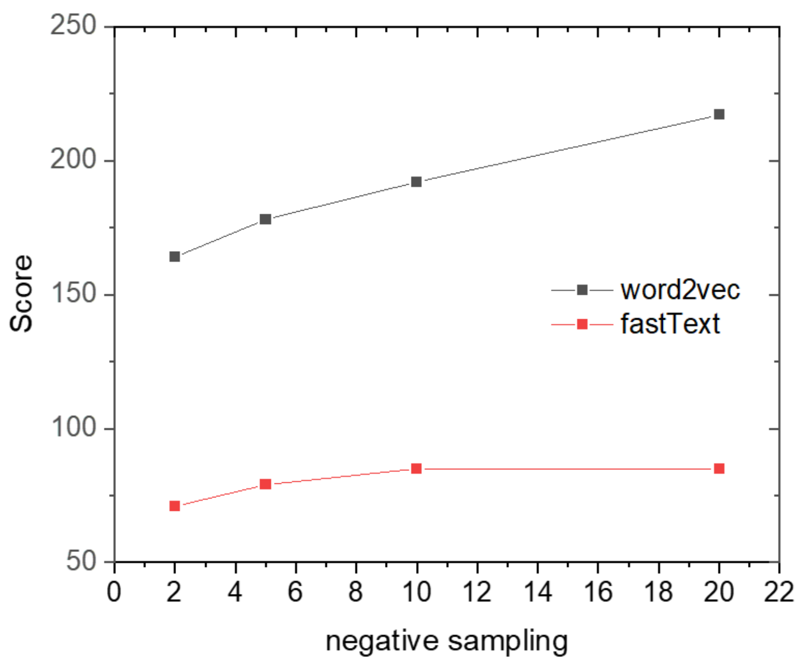

Figure 4.3 Scores of the trained word2vec and fastText models with negative sampling of 2, 5, 10 and 20.

Mikolov et al. has reported in their original paper that the number of negative samples in the range 5-20 is useful for small training datasets, while selecting 2-5 negative words works well for large training datasets [6]. Figure 4.3 shows that the increased negative sampling significantly improves the word2vec model performance and achieves higher scores. On the other hand, it slightly increases the scores of the fastText model. The effect of increased negative sampling on the fastText model performance is minor.

4.2 Contextual language model – BERT

The BERT language model is fine-tuned on a pre-trained SciBERT model with the collected paper abstracts from Scopus. The model training time is 56.22 mins. Different layers of a BERT model encode very different kinds of information [23]. The strategy of feeding word embeddings in different layers depends on the specific application. Therefore, it is advisable to test performance of different pooling methods and choose the appropriate one.

The word embeddings in different layers of the BERT model are used to evaluate the model performance. The model performance is scored using the test dataset discussed in Chapter 3. The results are shown in Figure 4.4.

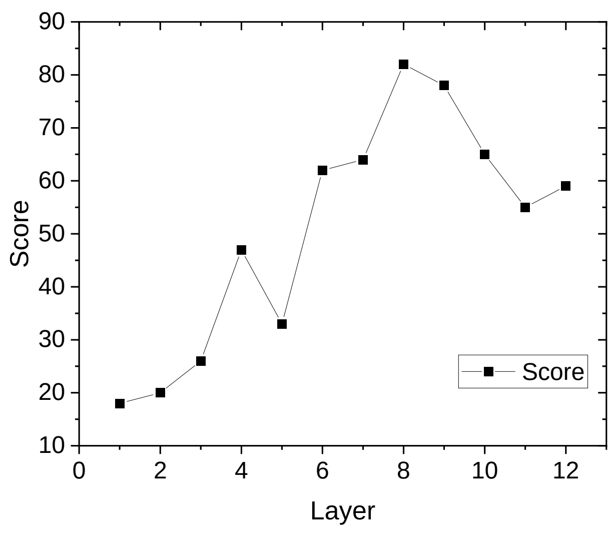

Figure 4.4 Scores of the word embeddings in different layers of the BERT language model.

The word embeddings of a BERT model do not show contextual information in the first layer. More and more contextual information is collected with each layer when the word embeddings move deeper into the network. However, the word embeddings in the final layer pick up task-specific information. For example, the word vectors in the final layer of this fined-tuned BERT model encode information used to determine a missing word. The reason is that a Masked Language Model head is applied while fine-tuning the BERT model. For the specific application in this work, the word embeddings in the Layer 8 exhibit the best result. Therefore, they will be used in the subsequent model optimisation process.

4.2.1 BERT Model Optimisation

The BERT model is optimised with various learning rates, i.e., 2e-5, 3e-5, 4e-5 and 5e-5. The batch size of 8 and epochs of 5 are used in the optimisation process. The effect of learning rate on the model performance for this specific application is shown in Figure 4.5. The initial learning rate of 2E-5 achieves the highest score in this specific application.

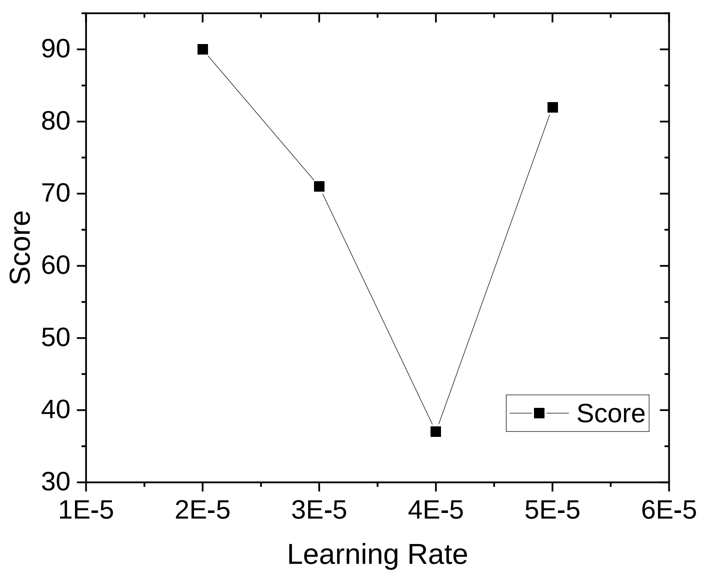

Figure 4.5 Effect of learning rate on the testing scores of the BERT model.

Chapter 5 Application

5.1 Knowledge Diagram

The word vector space is visually explored using Principal component analysis (PCA). Visual exploration is a crucial technique in data-driven applications. However, the trained word vector has a high dimension distribution, e.g., 100, 200, 300 etc, which makes it challenging to visualise the word vector distribution in a high dimension space. PCA is a dimension reduction technique whilst retaining most information. This is achieved using eigenvalues and eigenvectors of a data-matrix to provide less variables to represent maximum amount of distribution variation of the original data. Therefore, PCA makes feasible the important task of data visualisation. Figure 5.1 shows a 3D PCA plot of some selected concepts and materials in the GloVe language model (vector dimension size of 300, window size 30).

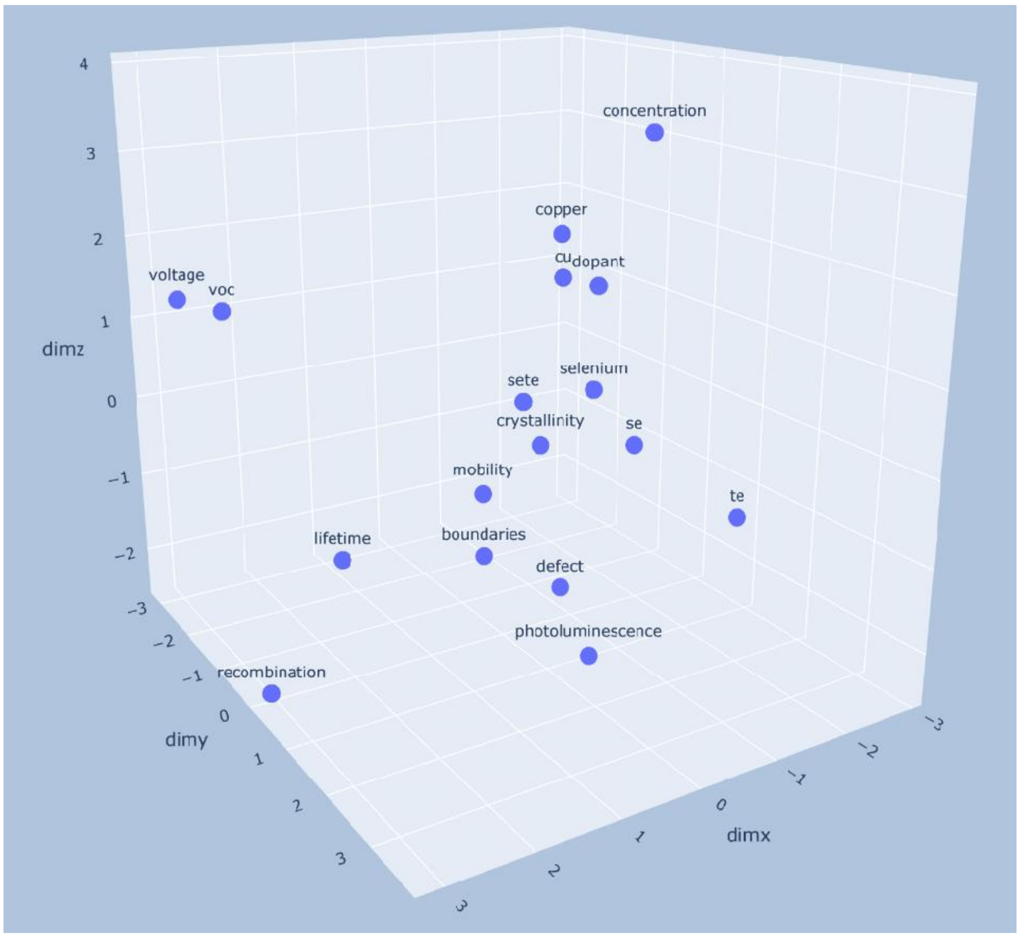

Figure 5.1 3D PCA plot of selected word vectors in the GloVe model.

From the 3D PCA plot, the interchangeable materials and concepts, such as "copper" and "Cu", "selenium" and "Se", "voltage" and "V_{oc}", stay close in the vector space. These interchangeable materials and concepts are among the topmost similar words of each other. The trained word embeddings also capture analogies between words. For example, "Cd – CdTe +ZnTe = ?", the calculated results are:

```
[('zn', 0.5797739028930664),
('te', 0.5100134611129761),
('m45', 0.49154233932495117),
('selfdiffusion', 0.48819923400878906),
('hg084zn016', 0.47685644030570984),
('featureless', 0.4641359746456146)]
```

"zn" is the first most similar word in the list. The accuracy is quite good.

The trained GloVe model outputs information-dense word embeddings, which contain valuable knowledge about the materials and relationships between them. A knowledge diagram is constructed in the vector space by connecting the most similar entities, i.e., materials and concept. Figure 5.2 shows the relationship between selected materials and concepts. The cosine distance is used to calculate the similarity between materials and concepts. The similarity can be ranked to indicate the likelihood that the materials will co-occur within a similar context. When a word appears in the topmost 50 similar words of another word, a solid line is used to connect these two words in the knowledge diagram. The relationship between materials and concepts can be deduced from the knowledge diagram. For example, concentration – lifetime – V_{oc}, carrier concentration is related to carrier lifetime because the activation ratio of dopants determines the carrier concentration and affects carrier lifetime. Carrier lifetime is a significant property to determine the open circuit voltage (V_{oc}) of CdTe solar cells. Therefore, adjusting the carrier concentration influences V_{oc}.

The similar materials, which are not explicitly reported in the literature, are potentially the predicted material candidates and need further experimental investigation. In conclusion, the trained embedding model effectively assimilates existing knowledge and the series of connections between materials and concepts can provide new insights to choose the most promising way forward to accelerate research of CdTe photovoltaics.

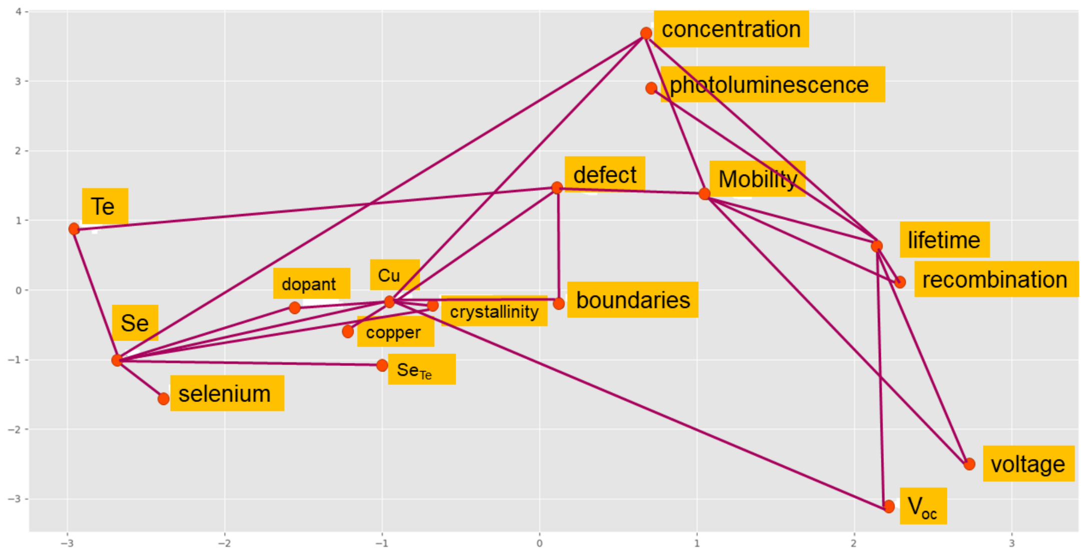

Figure 5.2 Knowledge diagram of selected concepts and materials.

5.2 Material Application Timeline

The language models are capable of tracking material applications. The CdTe dataset is split into three periods, i.e. 1909~1990, 1991~2010, 2010~2020. The choice of these three periods depends on the achieved conversion efficiency milestones of CdTe photovoltaic devices [24]. The respective GloVe language models are trained on the three CdTe datasets with a window size of 30 and a vector dimension size of 300. The top 500 most similar words are used to tack materials relevant to “defect”. The selected relevant materials to “defect” are shown Figure 5.3.

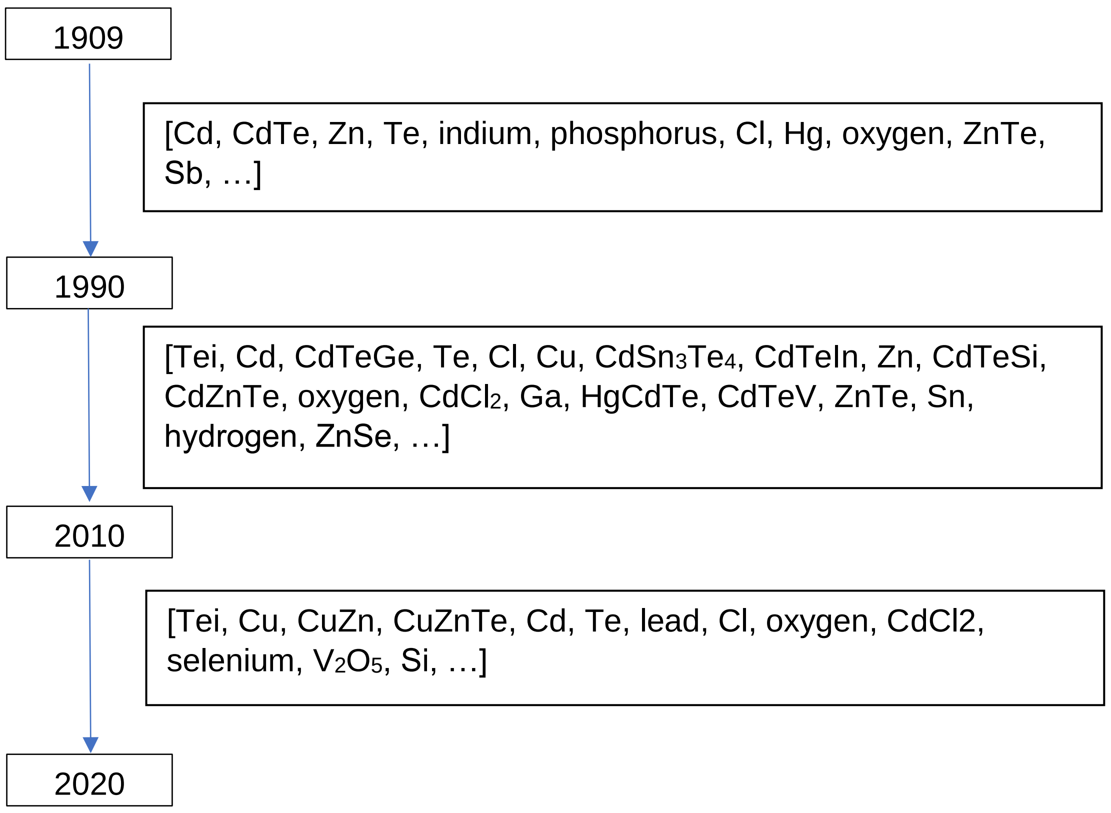
1909
[Cd, CdTe, Zn, Te, indium, phosphorus, Cl, Hg, oxygen, ZnTe, Sb, …]
1990
[Tei, Cd, CdTeGe, Te, Cl, Cu, CdSn3Te4, CdTeIn, Zn, CdTeSi, CdZnTe, oxygen, CdCl2, Ga, HgCdTe, CdTeV, ZnTe, Sn, hydrogen, ZnSe, …]
2010
[Tei, Cu, CuZn, CuZnTe, Cd, Te, lead, Cl, oxygen, CdCl2, selenium, V2O5, Si, …]
2020

Figure 5.3 Timeline of the selected most similar materials to "defect".

Oxygen is observed to appear in these three periods. Between 1909 and 1990, oxygen was reported to passivate defect-induced surface states on the valence band side of CdTe [25]. Between 1991 and 2010, oxygen was reported to induce isoelectronic oxygen-cadmium vacancy pair $(O_{Te}–V_{Cd})^{-/2-}$ during single crystalline CdTe growth [26]. Between 2011 and 2020, it was reported that careful control of oxygen in CdTe reduces intrinsic defects associated with cadmium and tellurium vacancies, interstitials, anti-sites in the bulk as well as within the grain boundary regions [27, 28]. Selenium is also shown in the top 500 most similar words to "defect" between 2011 and 2020. Selenium was demonstrated to passivate critical defects in the CdTe bulk to fabricate highly efficient CdTe solar cells [4]. A density functional theory calculation also showed that selenium passivates non-radiative recombination centres in CdTe [29]. Tracking materials in the most similar words provides a useful tool to discover promising new material applications. The appearance of "selenium" in the top 500 most similar words to "defect" corresponds to a published paper, which explicitly discussed the selenium passivation effect in CdTe [4]. The new materials in the most similar words potentially indicate new applications and need further comprehensive literature review.

Chapter 6 Conclusions and Future work

In this work, the application of natural language processing technologies on the research of CdTe solar cells is discussed. The language models of word2vec, GloVe, fastText and BERT are trained and evaluated using a custom test dataset. The training dataset includes more than 22,500 paper abstracts collected from the Scopus database. Before training the language models, the training dataset is pre-processed using tokenisation, converting all text to lower case, removing punctuation, removing stop words and lemmatisation. The custom test dataset consists of relevant material and concept word pairs in the field of CdTe solar cells.

The word vector dimension size, context window size and negative sampling are investigated in the training of word2vec, GloVe and fastText models. The BERT model is optimised with the learning rate. The word embeddings in each layer of the BERT model are used to test the model performance. After optimisation, the GloVe model achieves the highest score with the custom test dataset.

The word embeddings in the optimised GloVe model are exploited to construct a knowledge diagram and track the material application timeline. From the connections across the materials and concepts, we can obtain useful insights for future investigations.

The application of machine learning technologies on the material discoveries has just begun to be explored. The space of potential new applications is quite large. Substantial further improvements are necessary to accelerate the research of material science. The future research can be conducted using interdisciplinary approach. A training dataset will cover the research fields of solar cells, optoelectronics and photocatalysts. The interactions of relevant materials in these research fields will identify new pathways to highly efficient solar cells in future research.

References

1. EPIA, *solar powered growth in the UK.* 2014.
2. *Renewables 2021 Global Status Report*. 2021; Available from: http://www.ren21.net/.
3. Green, M., E. Dunlop, J. Hohl-Ebinger, M. Yoshita, N. Kopidakis, and X. Hao, *Solar cell efficiency tables (version 57).* Progress in Photovoltaics: Research and Applications, 2021. **29**(1): p. 3-15.
4. Fiducia, T.A.M., B.G. Mendis, K. Li, C.R.M. Grovenor, A.H. Munshi, K. Barth, W.S. Sampath, L.D. Wright, A. Abbas, J.W. Bowers, and J.M. Walls, *Understanding the role of selenium in defect passivation for highly efficient selenium-alloyed cadmium telluride solar cells.* Nature Energy, 2019. **4**(6): p. 504-511.
5. G. Ceder, a.K.P. *How supercomputers will yield a golden age of materials science*. 2013; Available from: http://www.scientificamerican.com/article/how-supercomputers-will-yield-a-golden-age-of-materials-science/.
6. Mikolov, T., I. Sutskever, K. Chen, G.S. Corrado, and J. Dean, *Distributed Representations of Words and Phrases and their Compositionality.* ArXiv, 2013. **abs/1310.4546**.
7. Tshitoyan, V., J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K.A. Persson, G. Ceder, and A. Jain, *Unsupervised word embeddings capture latent knowledge from materials science literature.* Nature, 2019. **571**(7763): p. 95-98.
8. Patel, P. and S.P. Ong, *Artificial intelligence is aiding the search for energy materials.* MRS Bulletin, 2019. **44**(3): p. 162-163.
9. Kim, E., K. Huang, A. Saunders, A. McCallum, G. Ceder, and E. Olivetti, *Materials Synthesis Insights from Scientific Literature via Text Extraction and Machine Learning.* Chemistry of Materials, 2017. **29**(21): p. 9436-9444.
10. Jensen, Z., E. Kim, S. Kwon, T.Z.H. Gani, Y. Román-Leshkov, M. Moliner, A. Corma, and E. Olivetti, *A Machine Learning Approach to Zeolite Synthesis Enabled by Automatic Literature Data Extraction.* ACS Central Science, 2019. **5**(5): p. 892-899.
11. Pennington, J., R. Socher, and C. Manning. *GloVe: Global Vectors for Word Representation*. in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2014. Doha, Qatar: Association for Computational Linguistics.
12. Bojanowski, P., E. Grave, A. Joulin, and T. Mikolov, *Enriching Word Vectors with Subword Information.* Transactions of the Association for Computational Linguistics, 2017. **5**: p. 135-146.
13. Rajapaksha, I. *BERT Word Embeddings Deep Dive*. 2020; Available from: https://medium.com/analytics-vidhya/bert-word-embeddings-deep-dive-32f6214f02bf.
14. Collobert, R. and J. Weston, *A unified architecture for natural language processing: deep neural networks with multitask learning*, in *Proceedings of the 25th international conference on Machine learning*. 2008, Association for Computing Machinery: Helsinki, Finland. p. 160–167.
15. Mikolov, T., S. Kombrink, L. Burget, J. Černocký, and S. Khudanpur. *Extensions of recurrent neural network language model*. in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2011.
16. Mikolov, T., K. Chen, G.S. Corrado, and J. Dean. *Efficient Estimation of Word Representations in Vector Space*. in *ICLR*. 2013.
17. Rong, X., *word2vec Parameter Learning Explained.* ArXiv, 2014. **abs/1411.2738**.
18. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin, *Attention is All you Need.* ArXiv, 2017. **abs/1706.03762**.
19. Mayo, M. *A General Approach to Preprocessing Text Data*. Available from: https://www.kdnuggets.com/2017/12/general-approach-preprocessing-text-data.html.
20. Google. *BERT*. Available from: https://github.com/google-research/bert.
21. Ethayarajh, K., *How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings.* ArXiv, 2019. **abs/1909.00512**.

22. Ringnér, M., *What is principal component analysis?* Nature Biotechnology, 2008. **26**(3): p. 303-304.
23. Alammar, J. *The Illustrated BERT, ELMo, and co. (How NLP Cracked Transfer Learning)*. Available from: http://jalammar.github.io/illustrated-bert/.
24. NREL. *Best Research-Cell Efficiency Chart*. [cited 2021 21 March]; Available from: https://www.nrel.gov/pv/cell-efficiency.html.
25. Orlowski, B.A., J.P. Lacharme, S. Bensalah, and C.A. Sebenne, *Electronic properties of cleaved CdTe(110) surfaces.* Surface Science, 1988. **200**(1): p. L460-L464.
26. Awadalla, S.A., A.W. Hunt, K.G. Lynn, H. Glass, C. Szeles, and S.-H. Wei, *Isoelectronic oxygen-related defect in CdTe crystals investigated using thermoelectric effect spectroscopy.* Physical Review B, 2004. **69**(7): p. 075210.
27. Gessert, T.A., S.H. Wei, J. Ma, D.S. Albin, R.G. Dhere, J.N. Duenow, D. Kuciauskas, A. Kanevce, T.M. Barnes, J.M. Burst, W.L. Rance, M.O. Reese, and H.R. Moutinho, *Research strategies toward improving thin-film CdTe photovoltaic devices beyond 20% conversion efficiency.* Solar Energy Materials and Solar Cells, 2013. **119**: p. 149-155.
28. Flores, M.A., W. Orellana, and E. Menéndez-Proupin, *First-principles DFT +G W study of oxygen-doped CdTe.* Physical Review B, 2016. **93**: p. 184103.
29. Watts, M.J., T.A.M. Fiducia, B. Sanyal, R. Smith, J.M. Walls, and P. Goddard, *Enhancement of photovoltaic efficiency in CdSe (x) Te(1-x) (where 0 ⩽ x ⩽ 1): insights from density functional theory.* J Phys Condens Matter, 2020. **32**(12): p. 125702.

Appendix A

A.1 Programming snippet of text pre-processing

```
def preprocess(text):

    # convert text into lower case
    Lower_case = [word.lower() for word in text]

    # remove punctuation
    punct_str = '!"#$%&\'()*+,./:;<=>?@[\]^_`{|}~'
    text_p = "".join([char for char in lower_case if char not in punct_str])

    # tokenisation
    words = word_tokenize(text_p)

    # remove stop words
    stop_words = stopwords.words('english')
    filtered_words = [word for word in words if word not in stop_words]

    # lemmatisation
    lemmatizer = WordNetLemmatizer()
    lemmas = [lemmatizer.lemmatize(word, pos='v') for word in filtered_words]

    return lemmas
```

Appendix B

B.1 Similarity Test Dataset

Word 1	Word 2
Se	passivate
Se	selenium
Cu	p-type
Cu	p-cdte
Cu	dopant
Cu	defect
Cu	instability
Cu	degradation
arsenic	defect
passivate	recombine
defect	disorder
defect	vacancy
defect	trap
defect	midgap
defect	subgrain
defect	antisite
defect	void
mobility	resistivity
mobility	conductivity
shunt	fill-factor
shunt	rollover
barrier	contact
barrier	crossover
barrier	cu
concentration	dopant
alignment	interfaces
alignment	photovoltage
rollover	ideality
rollover	fill-factor
recombination	trap
recombination	passivation
recombination	defect
recombination	barrier
recombination	midgap
recombination	dlts
recombination	grain-boundaries
recombination	photogeneration

recombination	vbo
recombination	jo
growth	nucleation
growth	crystallization
growth	epitaxy
growth	morphology
growth	roughness
growth	anneal
cdcl2	boundaries
cdcl2	anneal
dopant	cu
morphology	sem
morphology	texture
transmission	spectrophotometer
roughness	afm
crystallinity	texture
crystallinity	xrd
bandgap	se
trap	midgap
trap	defect
trap	recombination
trap	fermi
trap	activation
trap	mobility
trap	jo