GHENT UNIVERSITY

Natural language processing for materials science: an exploration

Author:
Fleur HUBAU

Supervisors:
Prof. Dr. Stefaan COTTENIER
Dr. ir. Michael SLUYDTS

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Physics and Astronomy

in the

FACULTY OF SCIENCES

Department of Physics and Astronomy

January 2021

This work has been performed at the Center for Molecular Modeling (CMM).

CENTER FOR MOLECULAR MODELING

Voorwoord

Ik heb altijd al gehouden van taal en wetenschap. Toen ik na mijn middelbare schooltijd voor de keuze werd gesteld welke weg ik op zou gaan, dan koos ik na lang wikken en wegen voor de richting die mijn verwondering het meest prikkelde: fysica. Fysicus word je echter niet zomaar. Het vergde veel inspanning en tijd om me de concepten uit de fysica eigen te maken. Net zoals een wielrenner traint voor de voorjaarsklassiekers, zo trainde ik langzaam het vermogen om te denken zoals een fysicus: abstract en rationeel, de randvoorwaarden indachtig. Hoewel boeiend en verrijkend, verliep het trainingsproces niet zonder slag of stoot. Vooral mijn eerste jaren aan de universiteit waren erg uitdagend. Naargelang ik vorderde doorheen de studie, ging het echter vlotter en voelde ik dat ik zowel op persoonlijk als op intellectueel vlak groeide. Het orgelpunt van dit alles is deze masterthesis, die zowaar mijn twee liefdes wist te combineren. Taal en wetenschap komen op een mooie manier samen in natural language processing (NLP). Wat NLP precies is en wat ik ermee aangevangen heb, kunt u lezen in dit werk, maar eerst wil ik graag nog enkele mensen bedanken.

Waar anders te beginnen dan bij mama en papa, de twee mensen die me gevormd hebben tot wie ik vandaag ben. Zonder jullie onvoorwaardelijke steun was ik nooit tot hier geraakt. Met mijn diploma (bijna) op zak kan ik eindelijk beginnen aan het volgende avontuur in mijn leven, maar wees gerust: jullie blijven altijd de veilige haven waar ik graag naar terugkeer. Aan jullie en ook aan Laurens, Celine en Brecht: bedankt om er altijd voor mij te zijn.

De promotoren van deze thesis, Stefaan Cottenier en Michael Sluydts, hebben me doorheen deze masterthesis uitstekend begeleid. Hoewel er geen fysiek overleg mogelijk was door de COVID-19 situatie, waren jullie toch beschikbaar op de momenten dat het nodig was. Onze online meetings hielpen me om het grote plaatje te bijven zien. Dankjulliewel voor de vele tips en de feedback die jullie me gaven, alsook voor het naleeswerk. Wanneer het soms wat moeizamer ging, kreeg ik telkens weer nieuwe inspiratie en motivatie door jullie oprechte betrokkenheid bij dit project.

Ook Umicore en het Centrum voor Moleculaire Modellering (CMM) wil ik graag bedanken. Samen vormden zij de aanleiding voor deze thesis. Ik hoop dat dit werk kan bijdragen tot verder onderzoek en toepassingen in de industrie.

Verder richt ik graag een woord van dank aan mijn voltallige VVN bestuur. De combinatie van VVN voorzitter en thesisstudent was bij momenten erg pittig, maar jullie uitstekende werk zorgde ervoor dat we ondanks de vleermuizen uit China toch enkele geslaagde (online) activiteiten op poten hebben gezet. Merci, team!

Afsluiten doe ik met de persoon die ik het liefste zie en liefst nog lang graag zal zien. Simon, je nuchtere kijk en bemoedigende woorden waren alles wat ik nodig had om de eindmeet van deze thesis te halen. Jij en ik, samen zijn we meer dan de som der delen.

Fleur Hubau

Gent, 18 januari 2021

Abstract

**Natural language processing
for materials science: an exploration**

by Fleur HUBAU

This master's thesis will explore the applicability of natural language processing (NLP) in materials science. NLP is widely used in today's world, with applications that include language translation, speech recognition, sentiment analysis, etc. More recently, NLP has made its entrance in the field of materials science, where it is applied to extract relevant information from scientific literature. Both supervised and unsupervised NLP can be distinguished. In supervised NLP, large hand-labeled datasets are used as training data to construct information-dense word embeddings, i.e. vector representations of words. Unsupervised NLP however, does not require human labeling. The potential of unsupervised NLP has been shown by Tshitoyan et al. in their paper "Unsupervised word embeddings capture latent knowledge from materials science literature"[1].

In this master's thesis, the pretrained word embeddings of the Mat2vec model by Tshitoyan et al. are used to assess the materials science knowledge that is captured in the embeddings. This is done for three different types of applications, in increasing order of complexity: (1) predicting properties of atoms and of elemental crystals, using the embeddings of chemical element names and a linear regression model, (2) predicting the formation energy of quaternary crystals, using a neural network, and (3) exploring an in-house database to extract promising candidates for thermoelectric materials. In all cases, we will discuss to what extent the word embeddings do or do not facilitate these tasks.

Nederlandse Samenvatting

Computers die moeiteloos kunnen converseren met mensen, ze bestaan. Althans in sciencefiction films. In de echte wereld zijn we nog ver verwijderd van pratende computers zoals Tony Starks J.A.R.V.I.S. of HAL 9000, de opstandige boordcomputer uit '2001: A Space Odyssey'. Het vakgebied dat hier iets aan wil veranderen is natural language processing of kortweg NLP. NLP houdt zich bezig met het aanleren van menselijke taal aan computers, zodat ze in staat zijn om taal te begrijpen en interpreteren zoals mensen dat kunnen.

Je staat er waarschijnlijk niet bij stil, maar de toepassingen van NLP in ons dagelijks leven zijn schijnbaar eindeloos. De spellingscorrector in tekstverwerkers, vertaalapplicaties (vb. Google Translate), virtuele (spraak)assistenten, de spamfilter van je e-mailprovider, de zoeksuggesties die je krijgt wanneer je iets opzoekt via een zoekmachine, chatbots... en ga zo nog maar een tijdje door. Het zijn allemaal voorbeelden van NLP die niet meer weg te denken zijn uit onze hedendaagse maatschappij.

Hoewel de spraakassistent in je smartphone er behoorlijk goed in slaagt om eenvoudige opdrachten zoals 'bel mama' of 'maak een afspraak in mijn agenda' uit te voeren, kan je voor een leuk babbeltje beter bij de buurman aankloppen. Waarom is het zo moeilijk om een computer te leren praten zoals een mens? Dat heeft alles te maken met de complexiteit van de menselijke taal en de vele nuances en ambiguïteiten die gesproken en geschreven tekst kleur geven. Een zin bevat vaak niet alle informatie die nodig is om hem te begrijpen, omdat een stukje van de betekenis vervat zit in de context. Ook veronderstelt taal een zekere kennis van de wereld om ons heen. De volgende voorbeelden brengen wat verduidelijking. Wanneer iemand zegt 'Toen de hamer op de glazen tafel viel, spatte hij uit elkaar', dan begrijp je als toehoorder dat het de tafel is die in duizend stukjes op de grond ligt, en niet de hamer. Een andere zin zoals 'Het meisje aaide de hond met de stok' vereist wat meer informatie: werd de hond, die een stok vasthad, geaaid door het meisje of aaide het meisje de hond met een stok? Een briefje van 'iemand die je leuk vindt' kan betekenen dat de briefschrijver een boontje heeft voor jou, maar omgekeerd kan ook. En zo zijn er nog talloze voorbeelden te bedenken. De boodschap is duidelijk: taal is vatbaar voor interpretatie.

Het menselijk brein kan rekenen op miljarden zenuwcellen (neuronen) die gespecialiseerd zijn in het ontvangen en doorgeven van informatie om taal op correcte wijze te interpreteren. Daarbovenop komen we al van jongs af aan in contact met gesproken en geschreven taal, waardoor ons brein uitgebreid de tijd heeft om de kunst van het spreken en begrijpen onder de knie te krijgen. Een computer beschikt jammer genoeg niet over zo'n handig menselijk brein. In de plaats gebruikt hij twee tekens om informatie op te slaan: 0 en 1, ook wel bits genoemd. Aangezien die bits met z'n tweeën zijn, zeggen we dat een computer een binaire machine is. Als we een computer taalvaardigheid willen bijbrengen, dan moeten we onze

woordenschat vertalen naar een vorm die leesbaar is voor de computer. Getallen worden in computers binair voorgesteld, dus als we alle woorden vertalen naar getallen zijn ze leesbaar voor de computer.

Woorden vertalen naar vectoren

Doorheen de geschiedenis van NLP zijn er een aantal methodes gebruikt om zo'n numerieke representatie van woorden te bekomen. In de traditionele NLP werd een reeks getallen gebruikt die overal nul was, behalve op één welbepaalde plaats. Die plaats was uniek voor elk woord. Zo'n representatie wordt ook wel een one-hot vector genoemd. Een voorbeeld illustreert deze methode het best. Stel dat we de eerste zin van 'De Hobbit', het fantastische fantasyboek van de Engelse schrijver J.R.R. Tolkien (1937), in one-hot vectoren willen gieten. De openingszin 'In een hol onder de grond woonde een hobbit' zou er dan als volgt uitzien.

$$\begin{aligned}
\text{in} &= [10000000] \\
\text{een} &= [01000000] \\
\text{hol} &= [00100000] \\
\text{onder} &= [00010000] \\
\text{de} &= [00001000] \\
\text{grond} &= [00000100] \\
\text{woonde} &= [00000010] \\
\text{hobbit} &= [00000001]
\end{aligned}$$

Elk uniek woord krijgt een vector, en omdat de zin 8 unieke woorden telt, heeft elke vector 8 plaatsen. Het eerste woord, 'in', krijgt op de eerste plaats een 1, alle andere plaatsen zijn ingenomen door nullen. Het tweede woord, 'een', krijgt een 1 op de tweede plaats in de vector en op alle andere plaatsen komt een nul te staan. Zo ga je door tot het woord 'hobbit', dat in zijn one-hot vector enkel op de laatste plaats een 1 krijgt. Op deze manier heeft elk woord een unieke vectorrepresentatie. Als we nu echter het volledige boek, dat meer dan 300 pagina's telt, in one-hot vectoren zouden vertalen dan krijgen we ongelofelijk lange vectoren die veel geheugen innemen en bovendien helemaal niet zo veel vertellen. Het is namelijk niet mogelijk om synoniemen of gelijkaardige woorden te herkennen door een reeks eentjes en nulletjes te vergelijken. Alle woorden zijn even verschillend van elkaar.

Er moest dus een efficiëntere manier gezocht worden om woorden naar getallen te vertalen zonder dat daarbij de betekenis verloren ging. Een manier die vandaag veel wordt gebruikt is Word2vec (letterlijk 'woord naar vector'). Het basisidee van Word2vec is dat de betekenis van een woord verweven zit in de woorden die het omringen. Dit wordt ook wel de context van een woord genoemd. Beschouw even opnieuw onze zin van daarnet: 'In een hol onder de grond woonde een hobbit'. Stel dat je niet weet wat een hobbit is, dan kan je toch uit de zin halen dat het iets is dat in een hol woont. Uit de context leiden we dus af dat het om een wezen of dier gaat. Verder leren we ook dat het woord 'hol' samengaat met het voorzetsel 'in'. Je kan dus 'in' een hol zitten en het hol bevindt zich 'onder' de 'grond'. De 'grond' is bijgevolg iets wat zich boven een 'hol' bevindt.

De context van een woord verschaft ons dus veel informatie over de betekenis of functie van een woord in een zin. Een computer kan deze informatie gebruiken om de betekenis van woorden te leren. Dat doen we door hem niet één, maar miljoenen zinnen te tonen die hij woord per woord zal bekijken. Een computermodel als Word2vec maakt bij het bekijken van de woorden gebruik van een kunstmatig neuraal netwerk. Zo'n netwerk bestaat uit kunstmatige neuronen die de 'echte' biologische neuronen uit het menselijke brein proberen na te bootsen. Het is in staat om te leren uit de verschillende contexten die het te zien krijgt. Door het kunstmatige neurale netwerk een hele hoop zinnen te voeden, zal het na verloop van tijd leren welke woorden vaak samen voorkomen of op dezelfde manier gebruikt worden. Het resultaat van dat leerproces zit vervat in de uiteindelijke woordvectoren, die gelijkaardig zijn voor gelijkaardige woorden. In Word2vec bestaan woordvectoren standaard uit 300 getallen. Dat is heel wat minder dan de one-hot vectoren van zonet, die evenveel plaatsen hadden als er unieke woorden waren.

Dat Word2vec onze intuïtie voor taal goed vat, kan visueel geïllustreerd worden door de woordvectoren op een tweedimensionaal vlak af te beelden. Op Figuur 1 worden enkele vrouwelijke en mannelijke woorden afgebeeld. Links zien we allemaal vrouwelijke woorden, rechts de mannelijke varianten. Het leuke is dat we ook kunnen rekenen met deze vectoren. Als je de vectoren voor 'koning' en 'man' aftrekt van elkaar en hierbij de vector voor 'vrouw' optelt, dan bekom je de woordvector voor 'koningin'. Mooi toch? Analoge berekeningen kunnen we doen met vectoren van bijvoorbeeld landen of bijvoeglijke naamwoorden. Vectorrekening met 'Parijs', 'Frankrijk' en 'België' resulteert in 'Brussel' en met 'groot', 'groter', 'klein' bekomen we 'kleiner'.

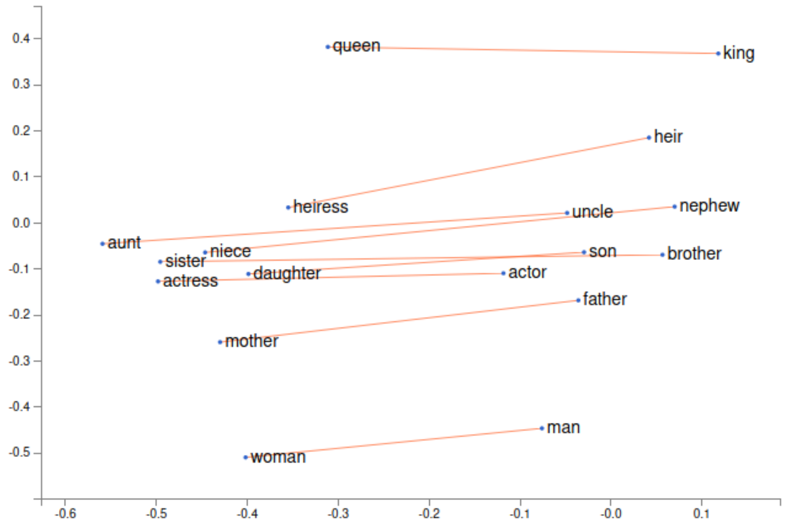


FIGURE 1: Een tweedimensionale voorstelling van de woordvectoren van uitgesproken vrouwelijke en mannelijke woorden[2].

NLP in de fysica

Missie geslaagd dus: we hebben een manier gevonden voor de computer om de betekenis van woorden te vatten. Maar wat heeft dit nu allemaal met fysica te maken? Heel veel, aangezien taal de manier is waarop wetenschappers in papers communiceren met elkaar. Het probleem van de laatste decennia is dat er, onder andere door de toenemende specialisatie in de wetenschappen, zo veel papers gepubliceerd worden dat niemand meer de tijd heeft om ze allemaal te lezen. Zo dreigt er belangrijke kennis verloren te gaan in een literatuurstapel die jaar na jaar groter wordt. In tegenstelling tot mensen kan een computer in enkele seconden de gehele literatuur van de laatste decennia doornemen en zo verbanden leggen die mensen er niet of slechts na enkele jaren van grondige literatuurstudie kunnen uithalen.

Vooral in de materiaalfysica werden met NLP al goede resultaten behaald. Een erg succesvol voorbeeld van NLP in materiaalfysica is 'Named Entity Recognition' (NER). Simpel gezegd maakt NER het mogelijk om woorden in een tekst te labelen. Aluminium, titaan en Fe krijgen bijvoorbeeld allemaal het label 'materiaal' opgeplakt. Woorden als geleidbaarheid, densiteit en poreusheid vallen onder de noemer 'materiaaleigenschap'. Door de woorden in een paper op deze manier te voorzien van labels, is het mogelijk om de tekst op een gestructureerde manier weer te geven. Voor één paper is dit niet echt nuttig, maar als we dit voor alle papers doen die tot nu toe in de materiaalfysica zijn verschenen, dan ontstaan er interessante mogelijkheden. Zo kan je uitgebreide zoekopdrachten uitvoeren over de gehele literatuur om te weten te komen wat de beste synthesemethode is voor een bepaald materiaal. Of indien je een nieuw soort batterij wil ontwikkelen bijvoorbeeld, dan kan je in een oogopslag te weten komen welke materialen in het verleden reeds getest werden en welke resultaten ze opleverden.

Ook wordt het mogelijk om metavragen te stellen. We zouden ons bijvoorbeeld kunnen afvragen welke materialen voor de meest diverse applicaties worden gebruikt. Het antwoord dat NER kan geven op deze vraag is afgebeeld in Figuur 2. In luik (a) zie je een 2D-projectie van de woordvectoren van 5000 woorden

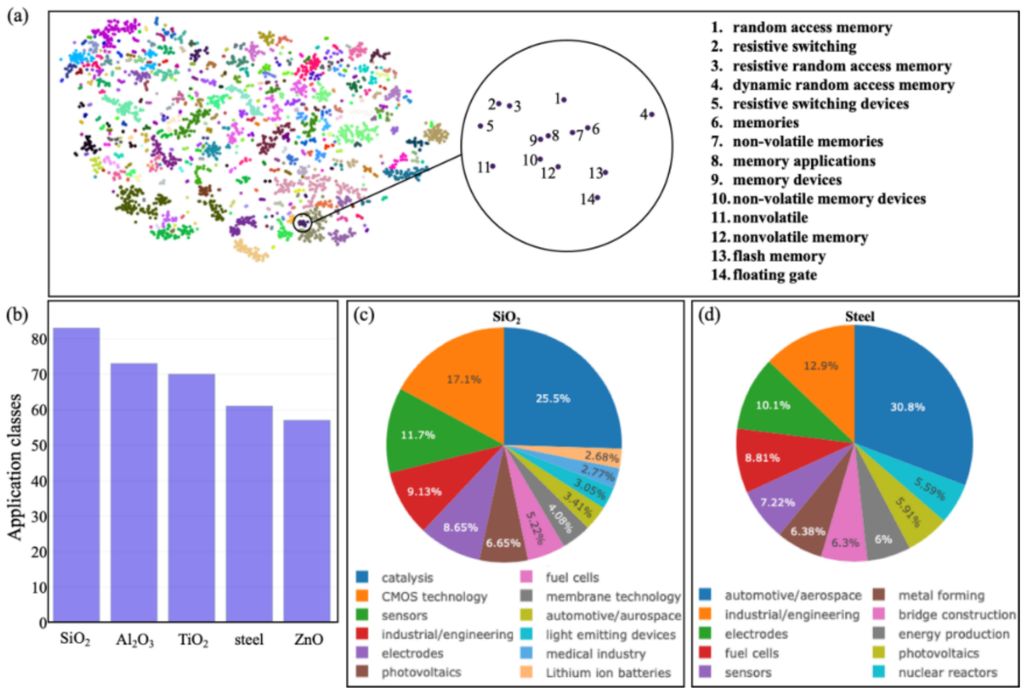


FIGURE 2: Een populaire NLP toepassing is 'Named Entity Recognition' (NER). NER kan metavragen zoals 'Welke materialen worden voor de meest diverse applicaties gebruikt?' beantwoorden[3].

die het label ‘toepassing’ opgeplakt kregen. De punten die dicht bij elkaar staan, vormen zogenaamde clusters en werden aangeduid met dezelfde kleur. Punten in dezelfde cluster hebben een gelijkaardige betekenis. Op de figuur zie je een uitgelicht voorbeeld van de toepassingen die onder de noemer ‘geheugen’ vallen. In paneel (b) is een histogram te zien van het aantal toepassingen per materiaal. Enkele van de hoogst gerangschikte materialen zijn SiO_2 en staal. In panelen (c) en (d) wordt een overzicht gegeven van de belangrijkste toepassingsgebieden van beide materialen. In één oogopslag hebben we zo een antwoord gekregen op een vraag die een materiaalwetenschapper slechts na enkele maanden zou kunnen beantwoorden.

Een gestructureerde representatie van tekst door NER maakt het leven van materiaalwetenschappers dus makkelijker, maar NLP kan nog veel meer betekenen voor de materiaalfysica. Zo kan de computer verbanden uit de literatuur halen die er niet letterlijk in vermeld staan. Dit is interessant wanneer we nieuwe materialen willen voorspellen voor bepaalde toepassingen. Zo toonde een belangrijke paper[1] aan dat nieuwe thermo-elektrische materialen[1] jaren voor ze ontdekt werden al voorspeld waren door een NLP model.

De NLP in materiaalfysica is dus niet enkel een handig hulpmiddel om inzicht te krijgen in de materialen die al onderzocht zijn, maar opent ook deuren naar nieuwe toepassingen van bestaande materialen. Het is duidelijk dat er nog veel werk nodig is vooraleer computers zullen praten zoals mensen, maar fysicapapers lezen kunnen ze al behoorlijk goed.

Doel en structuur van dit werk

Een algemene inleiding tot NLP wordt gegeven in Hoofdstuk 1. Ook wordt een kort overzicht gegeven van de huidige toepassingen van NLP in de materiaalfysica.

In Hoofdstuk 2 wordt gebruik gemaakt van de woordvectoren van chemische elementen om na te gaan in hoeverre zij nuttig zijn om bepaalde eigenschappen te voorspellen. Daarbij werd vertrokken van het volgende idee: als je enkel de woordvector van een chemisch element krijgt, kan je dan aan de hand van die vector achterhalen om welk element het gaat?

In Hoofdstuk 3 gaan we een stapje verder: daar draait het niet langer om enkelvoudige chemische elementen, maar om kristallen die opgebouwd zijn uit meerdere elementen. Meer bepaald gaat het over elpasolietkristallen. Dat zijn zgn. kwaternaire kristallen, omdat ze bestaan uit vier verschillende chemische elementen. De algemene formule van een elpasoliet is ABC_2D_6. Aan de hand van de woordvectoren van de A, B, C en D elementen en een neuraal netwerk werd geprobeerd om de vormingsenergie van de elpasolieten te voorspellen. De vormingsenergie is belangrijk in de materiaalfysica, omdat het ons veel kan leren over de stabiliteit van een materiaal.

Finaal wordt in Hoofdstuk 4 nagegaan of we op basis van woordvectoren nieuwe thermo-elektrische materialen kunnen voorspellen uit een gegeven database bestaande uit kwaternaire materialen.

[1]Thermo-elektrische materialen zijn materialen die heel efficiënt warmte naar elektrische stroom kunnen omzetten, waardoor ze erg interessant zijn voor warmterecuperatie.

Contents

List of Figures

List of Tables

Chapter 1

NLP and its significance in materials science

1.1 Introduction

The computer is incredibly fast, accurate, and stupid. Man is incredibly slow, inaccurate, and brilliant. The marriage of the two is a force beyond calculation.

Leo Cherne, economist

Since 1945, the number of published scientific articles and reports has doubled approximately every nine years[4]. This growing number of scientific publications, along with the continuing specialization of research domains, has made the ability for individual researchers to manually extract relevant information nearly impossible. What if the answers to all our questions are locked away in this enormous pile of literature? Sure, search engines and scientific literature databases can provide us with some answers when given the right keywords. However, nuanced questions such as 'Which piezoelectric material has the highest Curie temperature?' can not so easily be answered. What if we could teach computers to understand and interpret language just like humans do? This is exactly what natural language processing (NLP), a field of computer science and computational linguistics, is trying to do. NLP has many applications and is widely used today, both for spoken and written language. Some examples are given below [5, 6].

- Language translation, i.e. the translation of one language to another. Google translate, a free service from Google, is probably the best example of language translation. Users can input speech, text, images with text and even handwritten text that can be translated to more than hundred languages.
- Sentiment analysis, i.e. the characterization of subjective information such as opinions, communicated through text. Various companies rely on sentiment analysis to identify emotions in reviews or feedback from costumers. Satisfied or malcontent costumers provide valuable information for product analytics and market research.
- Spell checkers, software programs that identify and provide suggestions for incorrectly spelled words in a text, are widely used in word processors, email clients and search engines.

- Predictive typing, a technology that tries to predict text that users are inputting. For example in search engines, the next word in search queries is predicted, providing a drop-down list with suggested keywords.
- Speech recognition or speech-to-text is the conversion of spoken language to text. Today, virtual assistants from multinational technology companies such as Amazon (Alexa) and Apple (Siri) make use of speech recognition to interpret human speech and respond accordingly.

  Other examples of natural language processing include chatbots, email filters, automatic summarization of text, question answering... and many, many more. As you can see, NLP is everywhere and we use its applications daily without even thinking about it. However, just like learning a new language is difficult for any human, teaching a computer to process natural language is no easy task.

1.2 Word embeddings and Word2vec

You shall know a word by the company it keeps.

J.R. Firth, linguist

Natural language is a system created by humans to communicate with one another. This manner of human communication is incredibly nuanced and complex. Context, punctuation and intonation all add to the meaning of a sentence. For example 'Let's eat, grandma' is quite alarming without the comma and an ambiguous sentence such as 'He fed her cat food' does not sound pleasant either when it is wrongly interpreted. Luckily, the human brain can rely on billions of neurons to interpret spoken or written language. This is a very complex process that does not only require knowledge of vocabulary and grammar, but also the ability to guess what someone means based on the current situation, cultural differences, historical events, et cetera.

How can we teach this complex task to a computer? To begin with, we should somehow translate the meaning of words into something that a computer understands. In contrast to the neurons in human brains, computers use a binary system (based on 0 and 1) to store information. It thus seems that we have to convert words to numbers that a computer can store as binary data. Put differently, in order to make natural language comprehensible to a computer, each word in its vocabulary must be represented as a collection of numbers, which can be interpreted as the components of a vector. Some crucial questions immediately arise. What should word vectors look like in order to be useful for a computer and how can they be constructed?

1.2.1 One-hot encoding

Up until 2013[1], words in NLP models were regarded as discrete symbols and turned into vectors through one-hot encoding. The principle of one-hot encoding is best

[1]In fact, important work was done earlier by Bengio et al. in 2003[7] but this barely impacted the NLP community. The big paradigm shift in NLP came in 2013, when Mikolov et al. proposed the Word2vec algorithm as described in the next section (Section 1.2.2).

illustrated in an example. Consider the opening sentence of *The Hobbit*, the critically acclaimed fantasy novel by English author J.R.R. Tolkien (1932): 'In a hole in the ground there lived a hobbit'. The one-hot encoded word vectors that can be obtained from this sentence are given below.

$$\begin{aligned}
\text{in} &= [10000000] \\
\text{a} &= [01000000] \\
\text{hole} &= [00100000] \\
\text{the} &= [00010000] \\
\text{ground} &= [00001000] \\
\text{there} &= [00000100] \\
\text{lived} &= [00000010] \\
\text{hobbit} &= [00000001]
\end{aligned}$$

The number of components of each vector is equal to the number of unique words in the text, also called the vocabulary. Each index of the vector corresponds to a word. The first word has a 1 as its first member and the rest of the vector is put to zero. The second word has a 1 as its second member, the rest of the vector is zero, and so on for the other words. One can quickly understand the problems this approach brings: if we were to put the entire book of *The Hobbit* into one-hot encoded vectors the length of the vectors would be considerably large. This is bad news in terms of memory as a lot of space is wasted with zeros. Also, since these vectors are orthogonal, no similarity relationships can be derived between them. This is highly problematic if we truly want a computer to grasp the meaning of words.

1.2.2 Word2vec

A more efficient approach that encodes similarity in the vectors themselves was found in 2013 by Mikolov et al.[8] when the Word2vec algorithm was proposed. The general idea of Word2vec is that the meaning of a word is given by the words that frequently appear close-by, i.e. the word's context. The many contexts in which a word appears in a corpus of text are used to create a word vector, also called the embedding of the word. Rather than creating a word vector that represents the word itself, the aim is now to create a word vector that represents the environment of a word and therefore implicitly the word itself. Each word embedding is a vector of several hundred dimensions with non-zero elements. The word embeddings of words that appear in similar contexts are also similar. Hence, words that have the same meaning will have similar embeddings. When performing simple algebraic operations, we can obtain results that closely resemble the way humans interpret words. For example, subtracting the vector for 'man' from the vector for 'king' and adding the vector for 'woman' results in a vector that is closest to the vector for 'queen'[9].

A Word2vec model can be obtained by using one of two Word2vec architectures: the Continuous Bag-of-Words model (CBOW) or the Skip-gram model. Both architectures are artificial neural networks, meaning they are specifically constructed to mimic the neurons in a biological brain. They both compute word embeddings by looping over the words in a given corpus, one at a time. The Skip-gram architecture will use the current word in the loop to predict the words in its context. The CBOW architecture works the other way around. It predicts the current word in the loop based on the words in its context. The notion 'bag-of-words' refers to the fact that the order of words in the context is not important as their word vectors are averaged before being fed to the model[8]. In what follows, we will focus on the Skip-gram architecture as it best suits our purposes for this master's thesis. The idea behind Skip-gram is explained in detail in the next section.

1.2.3 The Skip-gram architecture of Word2vec

The Skip-gram Word2vec architecture is depicted in Figure 1.1. The current word

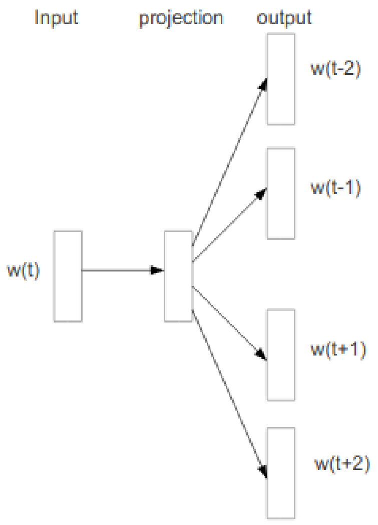


FIGURE 1.1: Schematic representation of the Skip-gram model[8].

w on position t has context words $w(t+1)$, $w(t+2)$, $w(t-1)$ and $w(t-2)$. The number of surrounding words that belong to the context of a certain word in addition to the word itself, is also called the window size. In this example, the window size is 5, but any number can be chosen[2].

More formally, the Skip-gram model tries to maximize the average log probability for the words on positions $t = 1, 2, ..., T$ in the training corpus. It does this based on the words that appear in a window size of c from a given word on position t[10]:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} \log p\left(w_{t+j} \mid w_t\right). \tag{1.1}$$

How does one calculate $p\left(w_{t+j} \mid w_t\right)$? We can do this by assigning two vectors to each word w: v when it is a center word and u when it is a context word. Then the probability that a context word O appears in the window size of the center word I is given by the normalized exponential function

$$p\left(O \mid I\right) = \frac{\exp\left(u_O^{\top} v_I\right)}{\sum_{w=1}^{W}\exp\left(u_w^{\top} v_I\right)}, \tag{1.2}$$

[2]Usually, the window size ranges from 2-10 words away. Increasing the window size improves the quality of the word embeddings, but also results in longer training time.

where W is the number of words in the vocabulary. In words, given a center word and a context word, we take the exponential of the dot product of the word vector of the center word with the word vector of the context word and normalize this result by performing a summation over the dot products of all the context vectors u_w in the vocabulary with the center word.

Equation 1.2, also called the softmax function, is frequently used in neural networks because it maps arbitrary values to a probability distribution. The exponential nature of the softmax function allows for assigning small probabilities to small values, while still amplifying the probabilities of the largest values. Physicists will recognize this function as the widely used Boltzmann distribution from thermodynamics.

1.3 NLP in materials science

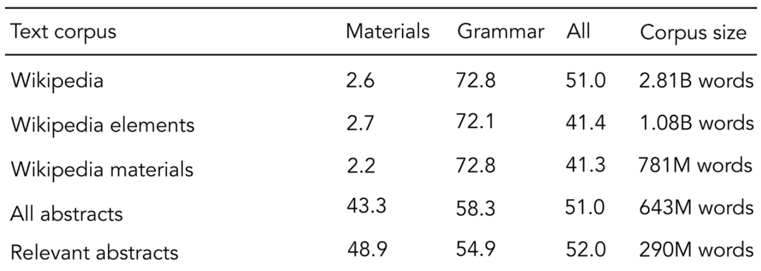
AI won't replace scientists, but scientists who use AI will replace those who do not.

Jeff Nichols, computational scientist

Over the past several years, the use of artificial intelligence (AI), and more specifically machine learning, has played an increasing role in materials science. Machine learning (ML) is a subset of AI that enables systems to learn from experience instead of being explicitly programmed to perform complex tasks. ML can be divided into two categories: supervised and unsupervised learning. As the name suggests, supervised ML requires human interference as it uses hand-labeled data to learn the relationship between an input X and an output Y. Unsupervised ML does not require labeled datasets. It is merely used to discover the underlying structure of data such as clustering and the detection of unusual data points. ML in materials science has various applications, such as new materials discovery and material property prediction[11].

More recently, also NLP has made its entrance in the field of materials science. In any NLP application for materials science purposes, words are converted to embeddings (see Section 1.2) using a large dataset of materials science articles. In practice, materials science abstracts are used, rather than full articles. Why is this? Generally, the information in abstracts is straightforward and contains few unnecessary words. Expanding the corpus with domain-irrelevant content, for example from Wikipedia articles, only creates more noise and is detrimental for the quality of the embeddings (see Figure 1.2) for materials science purposes. The scores on Figure 1.2 were obtained by assessing whether the NLP model could correctly finish grammar and science analogies such as 'Structure is to structures as energy is to ...' (the correct answer is of course 'energies') and 'He is to helium as Fe is to ...', with 'iron' as the expected answer. We also see on Figure 1.2 that irrelevant abstracts have to be discarded to improve performance for materials science NLP. This can be automatically done with a text classifier that solely selects abstracts that contain certain keywords and thus are expected to be relevant for the task at hand. This shows that the quality of the data is far more important than the quantity of the data. The more domain-specific the corpus, the better the results.

Just like ML, supervised and unsupervised methods can be distinguished. This is discussed in Sections 1.3.1 and 1.3.2.



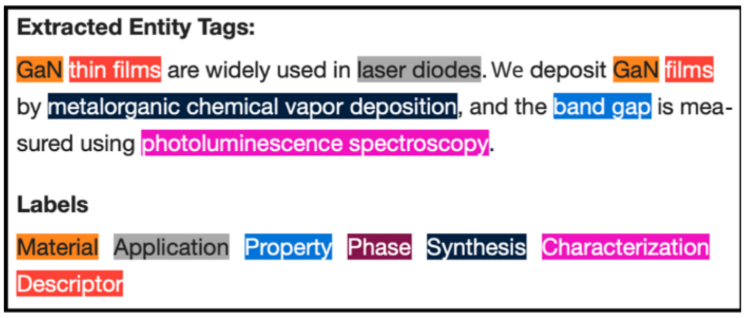
Text corpus	Materials	Grammar	All	Corpus size
Wikipedia	2.6	72.8	51.0	2.81B words
Wikipedia elements	2.7	72.1	41.4	1.08B words
Wikipedia materials	2.2	72.8	41.3	781M words
All abstracts	43.3	58.3	51.0	643M words
Relevant abstracts	48.9	54.9	52.0	290M words

FIGURE 1.2: The importance of corpus selection for NLP tasks in materials science. Models trained on Wikipedia articles, which contain the most text, perform very well for grammar analogies, but fall short when it comes to materials science analogies. It can also be seen that solely selecting relevant abstracts improves performance for NLP in materials science[1].

1.3.1 Supervised NLP

One important and widely-spread example of supervised NLP is Named Entity Recognition (NER). Originally, NER was intended for the extraction of names and geographic locations from unstructured text, but it soon found its applications in materials science. It can be used to convert large piles of unstructured text into structured database keywords such that quick information retrieval becomes possible.

A recent paper on NER [3] by Weston et al. illustrates how training a NER model works. First, seven entity labels were formulated: inorganic material (e.g. 'titania', 'Fe'), symmetry/phase label (e.g. 'tetragonal', 'fcc'), sample descriptor ('single crystal', 'nanotube'), material property (e.g. 'band gap', 'conductivity'), material application (e.g. 'photovoltaics', 'field-effect transistor'), synthesis method (e.g. 'solid state reaction', 'etching') and characterization method (e.g. 'XRD', 'photoluminescence'). With these entity labels, 800 materials science abstracts are hand-labeled by a materials scientist. Subsequently, a model is trained on millions of other abstracts to automatically recognize these entity labels in unstructured text. The result of a NER analysis can be seen in Figure 1.3. The NER model has indeed



Extracted Entity Tags:

GaN thin films are widely used in laser diodes. We deposit GaN films by metalorganic chemical vapor deposition, and the band gap is measured using photoluminescence spectroscopy.

Labels

Material Application Property Phase Synthesis Characterization Descriptor

FIGURE 1.3: Example of Named Entity Recognition (NER). The highlighted words are labeled with the corresponding entity name.

assigned an entity label where appropriate.

Weston et al. go on to show that quick literature search by means of some keywords becomes possible. For example, in the paper's accompanying web application Matscholar, https://www.matscholar.com/, a specific property or application entity can be entered, upon which all matching materials (along with the corresponding DOIs) will be returned. It is also possible to filter the search results, e.g. one can ask for all thermoelectric materials that do not contain lead (Pb). Another application that they mention is the automatic summarization of relevant information based on entities. One can enter a material and then quickly scan over the entities that co-occur most with the material of interest. The entities are ranked according to the number of occurrences with a given material. For example, one can learn how a specific material is commonly synthesized by looking at the highest-ranked synthesis entities and the application entities shed light on which applications a material is most associated with.

Furthermore, Weston et al. demonstrate that meta-questions, such as 'Which materials have the most diverse applications?', can be answered. Asking this question results in the answer depicted in Figure 1.4. In Figure 1.4 on panel (a),

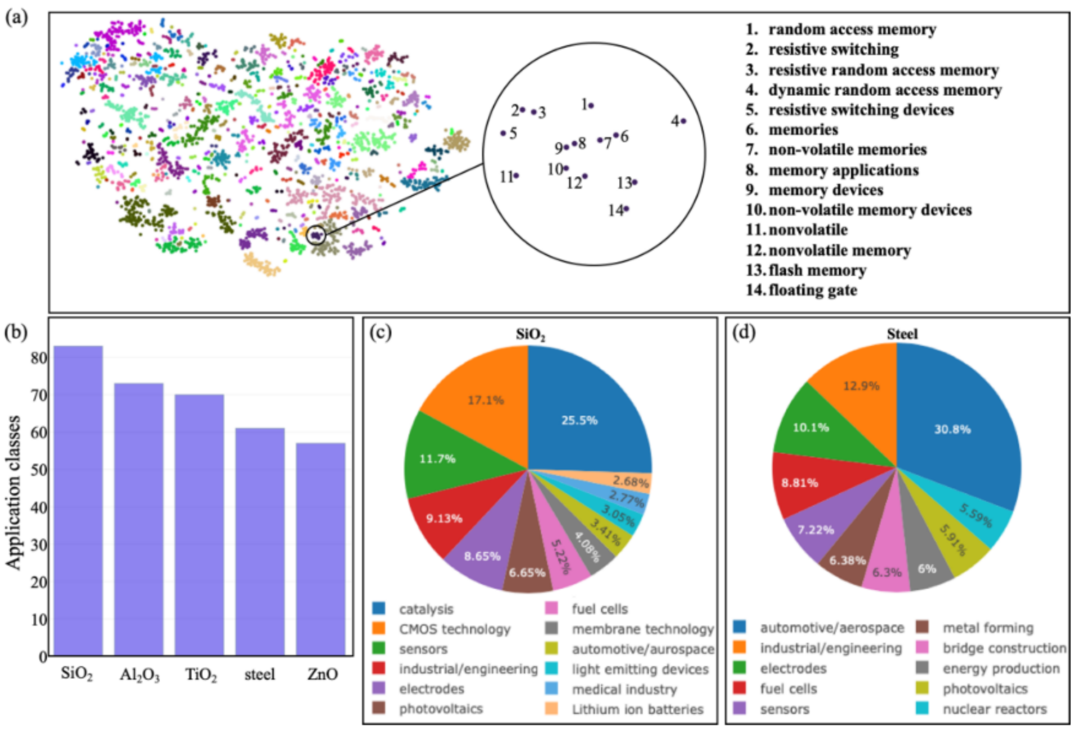


FIGURE 1.4: Answering the question 'Which materials have the most diverse applications?' with NER. Panel (a) shows a 2D-projection of the word embeddings from 5000 'application' entities. On panel (b), a histogram with the number of applications per material is given. Panels (c) and (d) illustrate the top applications for SiO_2 and steel in a pie chart[3].

one can see a 2D-projection of the word embeddings from 5000 'application' entities. The different colors refer to sub clusters with words that belong to the same category. The category 'solid-state memory devices' is used as an example on the figure. In Figure 1.4 (b), a histogram with the number of applications per material is given. Figures 1.4 (c) and 1.4 (d) show the top applications for two example materials (SiO_2 and steel) in a pie chart.

There are many other applications of the NER approach. The extraction of key experimental parameters for the synthesis of materials is probably the most important one. For example, the key synthesis parameters for 30 different oxide materials systems were determined using 76,000 articles that contained information on these oxides[12]. Another publication demonstrated that precursors for two perovskite materials could be predicted, by using training data that was published a decade before their first synthesis methods were even reported. The synthesis routes for new perovskite materials were also explored[13].

1.3.2 Unsupervised NLP

In contrast to supervised NLP methods which use hand-labeled training data, Tshitoyan et al. recently showed that the materials science knowledge in published literature can be contained in embeddings without human labeling[1]. Their unsupervised NLP approach resulted in the extraction of structure-property and chemical relationship information and also demonstrated that new thermoelectric materials can be discovered with word embeddings. Their Skip-gram Word2vec model that was obtained after a training procedure on millions of materials science abstracts was given the name Mat2vec.

The Mat2vec model

Tshitoyan et al. collected 3.3 million scientific abstracts from materials science articles, from which they obtained a vocabulary of $V = 500,000$ words. A word was included in the vocabulary when it occurred more than five times. Chemical formulae were added independent of the number of mentions, but were normalized such that the order of elements and common multipliers did not matter. For example, NiFe and $Fe_{50}Ni_{50}$ refer to the same word in the vocabulary. As can be seen on Figure 1.5, all words in the vocabulary are used in their one-hot encoded form: a V-dimensional vector with a 1 at their index and zeros everywhere else. The Skip-gram Word2vec algorithm then loops over all the one-hot encoded word representations. A neural network consisting of a single linear hidden layer predicts all words that appear in a chosen window size of the current word. Of course, while some words occur hundreds of times in the context of a certain word, others appear ever so slightly or not at all. This is reflected in the resulting word embedding of 200 dimensions[3] of a word. Finally, the softmax function (see section 1.2.3) normalizes the word embeddings to 1.

What is surprising, is that the obtained word embeddings capture chemical intuition, even though no chemical information was added to the model. This is illustrated when we perform some simple arithmetic on the word embeddings. For example, vector('BeO') - vector('Be') + vector('Mg') approximates the word vector of 'MgO'. The word embeddings also reflect the ferromagnetic/antiferromagnetic nature of NiFe and IrMn as vector('ferromagnetic') - vector('NiFe') + vector('IrMn') yields 'antiferromagnetic'.

Predicting thermoelectric materials

Tshitoyan et al. used the obtained embeddings of the Mat2vec model to predict new thermoelectric materials. Thermoelectrics are materials that efficiently convert heat into electric power, making them interesting candidates to recover waste heat for useful purposes. The performance of thermoelectric materials is captured by the

[3]Any number between 50 and 300 works well for the dimensionality of word embeddings, but the authors show that 200 was best in this case.

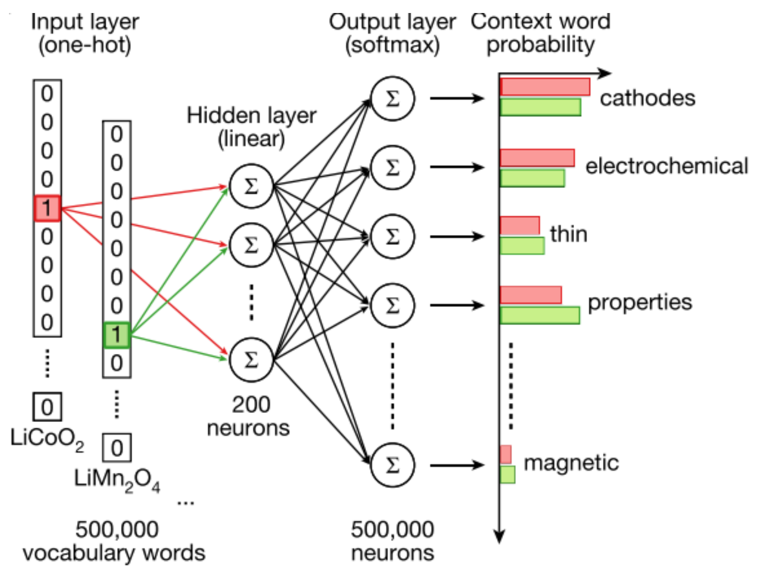


FIGURE 1.5: Schematic representation of the Skip-gram Word2vec algorithm by Tshitoyan et al. that was trained on 3.3 million scientific abstracts from materials science articles. The resulting model was named Mat2vec.

dimensionless figure of merit[14]

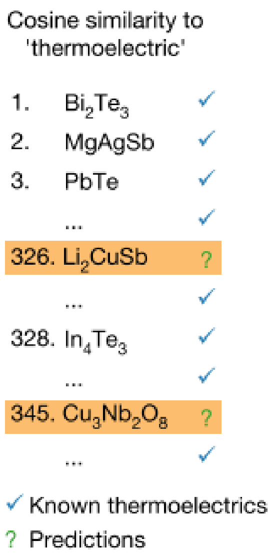
$$zT = \frac{\sigma S^2 T}{\kappa}, \tag{1.3}$$

with σ the electrical conductivity, S the Seebeck coefficient, T the temperature and κ the thermal conductivity. The factor σS^2 is also called the power factor and is more commonly used than the figure of merit zT for thermoelectrics.

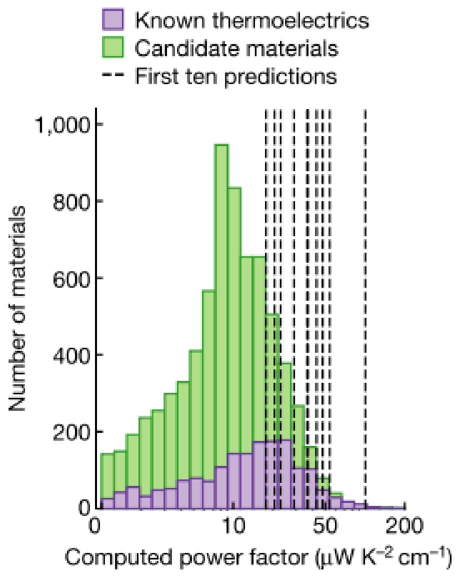
To find thermoelectric materials with the Mat2vec model, Tshitoyan et al. assessed which materials were most similar to the vector representation of the word 'thermoelectric'. This was done by calculating the cosine similarity of the word vector for 'thermoelectric' with the word vector of a certain material. The cosine similarity of two vectors **A** and **B** is defined as follows:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}. \tag{1.4}$$

The materials that yielded the highest cosine similarity are listed in Figure 1.6a. Apart from known thermoelectric materials such as Bi_2Te_3 and PbTe, we also see the appearance of materials that have never been studied as a thermoelectric material before. This is surprising, given the contextual nature of the embeddings. One might expect that a high cosine similarity with the word 'thermoelectric' only occurs for materials that are reported frequently as thermoelectric materials. However, the predictions that appear in Figure 1.6a have never been studied as thermoelectric materials, meaning they did not appear close to the word 'thermoelectric' or related words in any of the materials science abstracts. To investigate whether the predicted thermoelectrics are indeed promising thermoelectric materials, rather than false candidates, the authors looked at their thermoelectric power factor. The higher this thermoelectric power factor, the better suited a material is for thermoelectric applications. By comparing the known and the predicted thermoelectric materials with a dataset of computed thermoelectric power factors, a ranking was made of the materials for which computational data of the power factors were available (Figure 1.6b). The purple and green bars respectively represent the known thermoelectrics and the candidate thermoelectrics. The 10 highest ranked predictions[4] from the candidate materials are indicated by the dashed lines, yielding some interesting thermoelectric materials that were not discovered before for further experimental research.


Cosine similarity to 'thermoelectric'
1. Bi_2Te_3 ✓
2. MgAgSb ✓
3. PbTe ✓
... ✓
326. Li_2CuSb ?
... ✓
328. In_4Te_3 ✓
... ✓
345. $Cu_3Nb_2O_8$?
... ✓
✓ Known thermoelectrics
? Predictions

(A) Ranking of the highest cosine similarities of materials with the word 'thermoelectric'.



(B) Computed power factor for known and candidate thermoelectric materials.

FIGURE 1.6: Prediction of thermoelectric materials with the Mat2vec model[1].

As a means to generalize the predictive capabilities of the Mat2vec model, Tshitoyan et al. also performed what is best described as a historic search. They trained a Mat2vec model at various moments 'in the past' (between 2001 and 2018) with materials science abstracts up until that moment. Then they used these models to predict a top 50 of thermoelectric materials that were likely to be discovered in the future. Subsequently, they assessed what percentage of the predicted top 50 were years later published as newly found thermoelectric materials. Materials are branded as 'published thermoelectric material' when they appear alongside thermoelectric keywords such as 'zT', 'seebeck', 'thermoelectric', 'thermoelectrics', 'thermoelectrical', and so on.

The results are displayed in Figure 1.7. The grey lines indicate the cumulative percentage of predicted thermoelectric materials in the top 50 that were reported in the literature, years after the predictions were made. Let's take the year 2015 as an example. The Mat2vec model was trained with materials science abstracts before 1 January 2015. The grey line expresses the cumulative percentage of thermoelectrics in the predicted top 50 that were discovered after one, two, three and four years.

[4]For the interested reader, these are: Li_2CuSb, $CuBiS_2$, $CdIn_2Te_4$, $CsGeI_3$, $PdSe_2$, KAg_2SbS_4, $LuRhO_3$, MgB_2C_2, Li_3Sb and $TlSbSe_2$.

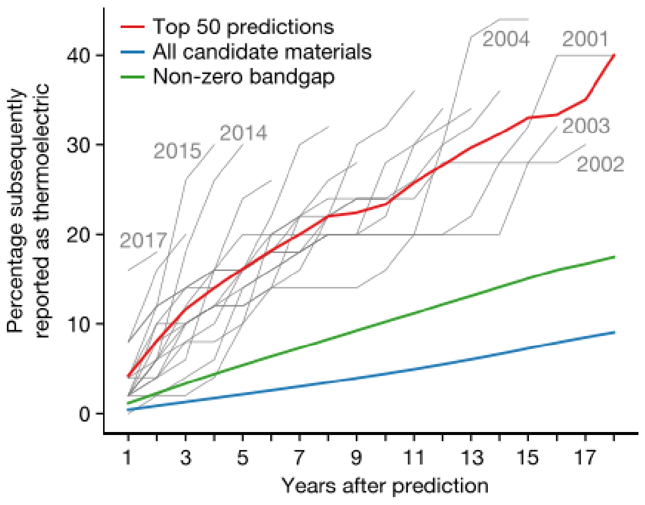


FIGURE 1.7: Prediction of top 50 thermoelectric materials using word embeddings obtained from various historical datasets. The grey lines indicate the cumulative percentage of predicted thermoelectric materials in the top 50 that were reported in the literature, years after the predictions were made. It is obvious that the average materials in the top 50 (red line) were more likely to be discovered compared to a randomly chosen material (blue line) or a random material with a non-zero band gap (green line).

Overall, we see that the predicted thermoelectrics in the top 50 are significantly more likely to be discovered (red line) compared to a randomly chosen material (blue line) or a random material with a non-zero band gap (green line)[5]. Notice that the grey lines of more recent years are steeper, implying that the use of larger corpora improves the number of successful predictions.

The authors performed a similar procedure with the words 'photovoltaics', 'topological insulator' and 'ferroelectric' and found trends similar to Figure 1.7.

1.4 Goal and structure of this work

We have seen in this chapter that NLP is widely used in today's world. Most importantly for this master's thesis, its increasing role in materials science has become apparent. Both supervised and unsupervised NLP approaches were considered in the previous sections. In this work, we will adapt the latter approach to find out what else is possible with the Mat2vec model from Tshitoyan et al. for materials science purposes. The structure of this work is listed below.

- Chapter 2: The embeddings of chemical elements in the Mat2vec model are used to create an overview of the two-dimensional embedding space. Furthermore, we investigate whether several atomic and thermodynamic properties can be predicted with embeddings and a linear regression model.
- Chapter 3: The formation energy of elpasolites (quaternary crystal structures with general formula ABC_2D_6) is predicted with a random forest and a neural network. The dataset that is used to train the models and to validate the results consists of the formation energy of approximately two million elpasolites. The

[5]Non-metals, which have a non-zero band gap, are known to have better thermoelectric properties.

aim of this chapter is to investigate whether the use of embeddings can prove to be helpful in a neural network to predict the formation energy of elpasolites.

- Chapter 4: An in-house database of the CMM is used to explore the potential of embeddings to predict thermoelectric materials. The results are compared to earlier work on the prediction of the thermoelectric power factor by Jasper De Witte.
- Chapter 5: Conclusion and outlook of this work.

Chapter 2

Property prediction for atoms and elemental crystals

An overview of some current NLP applications in materials science was provided in Chapter 1. Now we will apply these methods to problems we are interested in. A good place to start with is the embeddings of individual element names as they make up the materials and the structures that are the subject of study in materials science. All chemical elements that have been discovered thus far are included in the periodic table of the elements, which is depicted in Figure 2.1.

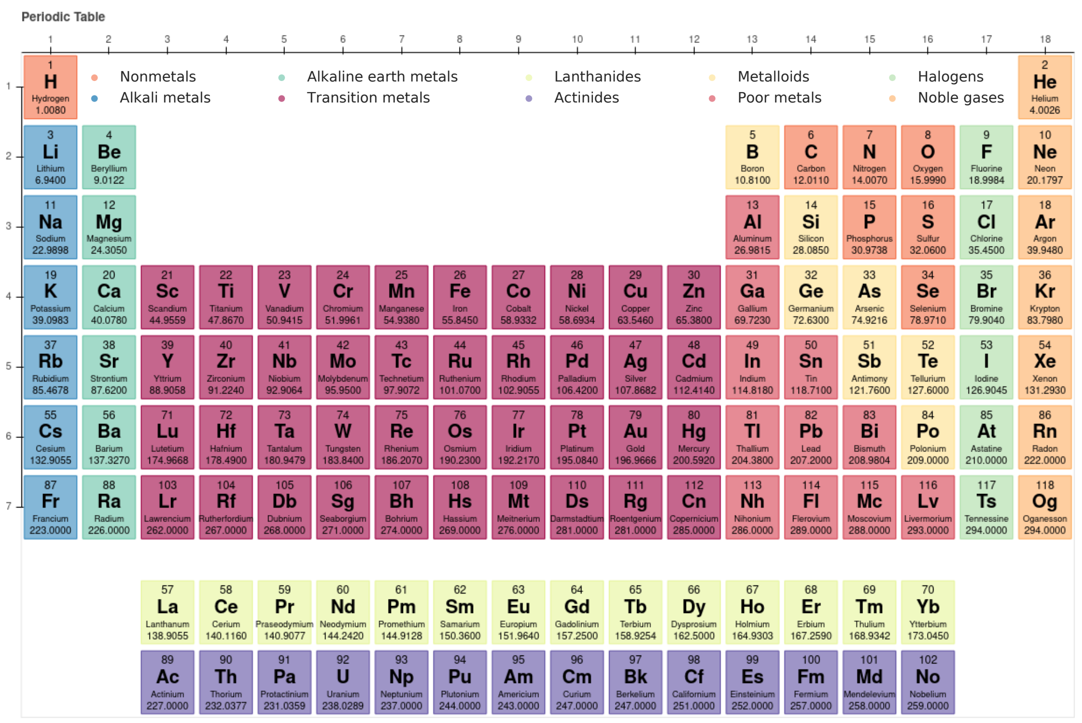


FIGURE 2.1: The periodic table of the elements[15]. The colours indicate the different categories into which the elements can be divided.

The elements in the periodic table are positioned by increasing order of atomic number, i.e. the total number of protons in the atomic nucleus. Elements in the same column (group) have similar properties as they have an equal number of electrons in their outermost shell, which determines their tendency to form chemical bonds with other atoms. The groups are numbered from 1 to 18. A row (period) in the periodic table contains elements that have an equal number of shells, increasing from 1 to 7. The various colours in the periodic table on Figure 2.1 correspond to the different categories into which the elements in the periodic table can be grouped together[1].

In the Skip-gram Word2vec algorithm on which Mat2vec is based, words that have similar meanings will also have similar embeddings as they will often be used in analogous contexts. When we look at the elements in the periodic table, we thus expect that elements from the same category will possess similar embeddings. To assess whether the word embeddings of the elements in the periodic table are indeed similar, we will project the 200-dimensional embeddings down to two dimensions in Section 2.1 to envision the word embedding space in which the chemical elements live.

Apart from a general overview of the elemental embeddings in the embedding space, it is also interesting to look at the individual embeddings in a more quantitative way. Given the embedding of a random chemical element, can we discover the name of that element by predicting several properties with its word vector? This is the general idea of Section 2.2. By predicting several atomic and thermodynamic properties with a linear regression model and the embeddings as input, the ability of word embeddings to capture chemical information will be assessed.

The pretrained Mat2vec word embeddings from the paper 'Unsupervised word embeddings capture latent knowledge from materials science literature'[1] are used in this chapter. As in the paper by Tshitoyan et al., the element's embeddings are retrieved by using the full element's name (e.g. 'hydrogen'), rather than its symbol ('H'). Exactly 100 elements and their corresponding 200-dimensional embedding vectors remain, after discarding the elements that were not present in the vocabulary of the Mat2vec model with their full element name[2].

2.1 Word embedding space of the elements

To get some visual insight in the relations between the embeddings of the elements, we will create a two-dimensional embedding space of the elements in the periodic table. In order to plot the embedding space of the elements onto a plane, we must first reduce the 200-dimensional embeddings of the chemical elements to two dimensions. This was done with the dimension reduction algorithm UMAP, which is short for 'Uniform Manifold Approximation and Projection'. In general, UMAP constructs a high dimensional graph representation of the data points, which is sort of a schematic representation of all the connections in the data, and then a low-dimensional graph is optimized to be as structurally similar as possible to the high dimensional graph. The interested reader is directed to the accompanying paper[16] for mathematical details.

The resulting two-dimensional embedding space after application of UMAP is depicted in Figure 2.2. It is important to note that a lot of information is lost due

[1]That is, according to the most commonly accepted version (Los Alamos National Laboratory, `https://periodic.lanl.gov/metal.shtml`).

[2]In total, 18 element names were discarded because they did not appear in the Mat2vec vocabulary. These element names are berkelium (Bk), einsteinium (Es), mendelevium (Md), nobelium (No), rutherfordium (Rf), dubnium (Db), seaborgium (Sg), bohrium (Bh), hassium (Hs), meitnerium (Mt), darmstadtium (Ds), roentgenium (Rg), nihonium (Nh), flerovium (Fl), moscovium (Mc), livermorium (Lv), tennessine (Ts), oganesson (Og).

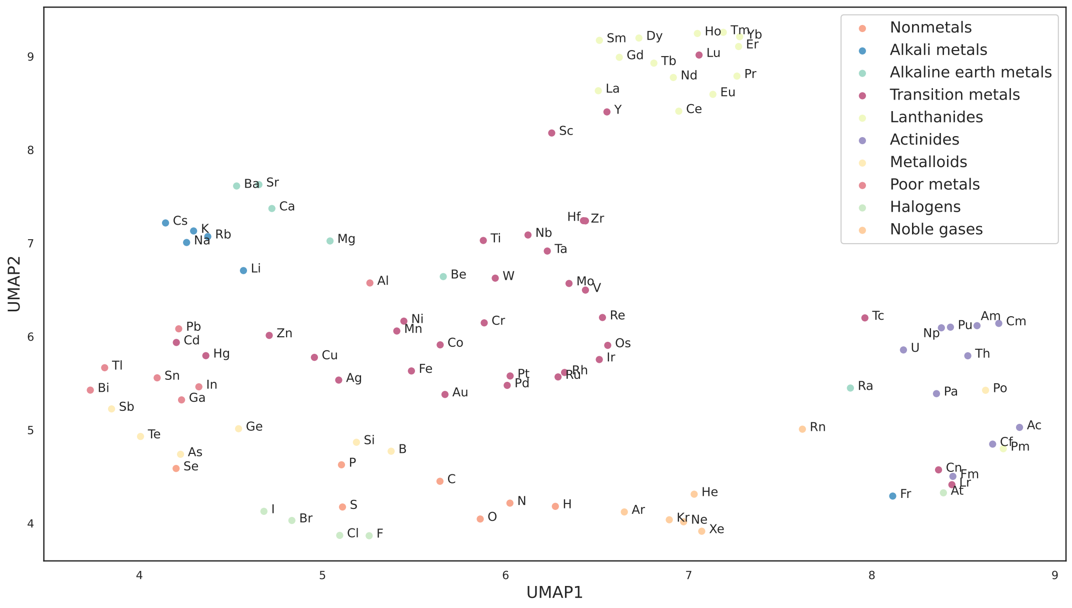


FIGURE 2.2: The embedding space of the elements in two dimensions after applying the dimension reduction algorithm UMAP to the 200-dimensional embeddings. The colour code is the same as in Figure 2.1.

to the dimensional reduction. The original 200-dimensional embeddings contain much more detail than a two-dimensional projection could ever provide us with. Nonetheless, Figure 2.2 illustrates that even the two dimensional embedding space contains some useful information. Generally speaking, it can be seen that the elements of the same category group together in the two-dimensional embedding space. Given that the word embedding space is a context-based representation, it makes sense that the noble gasses form a separate cluster. In contrast to the other elements, they are not frequently used in combination with other elements and this is reflected by the embeddings. Overall, the other categories group closer together. As is also pointed out in the paper by Tshitoyan et al., one can observe that the main components of organic compounds are clustered together: hydrogen (H), oxygen (O), nitrogen (N) and carbon (C) live in close proximity in the two-dimensional vector space. An other interesting observation can be made in the lower right corner of the embedding space. Radon (Rn), radium (Ra), technetium (Tc) and polonium (Po) all seem to be a little out of place there. However, as these are all radioactive elements, it comes as no surprise that they appear closer to uranium (U) and thorium (Th) than to the elements of their respective category.

In the following section, more than two dimensions will be taken into account to investigate the usefulness of the embeddings to predict certain elemental properties.

2.2 Properties from embeddings

The properties that are investigated here can be divided into two groups: properties that can be linked to the periodic table (atomic number, Mendeleev number, column and row in the periodic table, Pauling electronegativity and atomic weight) and thermodynamic properties (boiling point, evaporation heat, heat of formation and melting point). Notice that the individual element names can refer to elements as well as crystals. For example, with 'iron' (Fe) in the periodic table, the element iron with its atomic properties, such as atomic number and atomic weight, is meant. Thermodynamic properties (boiling point, evaporation heat...) are properties from iron compounds, i.e. structures that are made up from iron atoms, such as an iron crystal or liquid iron.

2.2.1 Approach

To access the properties of the elements, the python library mendeleev[17] is used. For all other tasks, we utilize the free software machine learning library scikit-learn[18].

The dimensionality of the embedding vectors first has to be reduced to avoid overfitting as we only have 100 data points at most per property, given we have 100 element names with their corresponding embeddings. Analogous to the approach of Tshitoyan et al., the 200-dimensional vectors are brought down to 15-dimensional vectors by applying principal component analysis (PCA)[19]. PCA reduces the dimensionality by identifying so-called principal components, which are capable of retaining (most of) the variation that was present in the original dataset. The principal components are linear combinations of the original variables. For our particular case, the resulting 15-dimensional embeddings after applying PCA are linear combinations of the original 200-dimensional embeddings and explain 65% of the total variance of the original embeddings. We thus unfortunately lose a substantial amount of information by applying PCA, but more than half is still conserved.

Subsequently, linear regression is performed with the 15-dimensional embeddings as features and a certain property of the chemical elements as output of the linear regression fit. The result of the linear regression procedure is a 15-dimensional vector $\vec{a}$ with the regression coefficients and an intercept value b. The linear regression model that is obtained is then used to predict the correct property value for each chemical element. This can be mathematically expressed as

$$p = \vec{a} \cdot \vec{w} + b, \tag{2.1}$$

where p is a certain property value, $\vec{a}$ is the vector that contains the regression coefficients for a given property, $\vec{w}$ is the 15-dimensional embedding vector of a chemical element and b is the intercept value of the regression fit.

For each property, we perform linear regression several times with k-fold cross-validation. For k-fold cross-validation, the total dataset is divided into k folds. A fold is a subsection of the dataset that contains $1/k$'th of the original data points in the dataset. The training set, which is the data that is used to build the linear regression model, is created by $k-1$ folds of the data. The validation set, which is the dataset that is used for predictions, consists of the remaining fold. This process is performed k times, such that every fold is used once as validation set. The implementation of k-fold cross-validation is important here because we have such a limited dataset. In other words, k-fold cross-validation results in less biased or less optimistic estimations of the linear regression procedure. A simple split in a training and validation set would possibly give more biased results.

Here, we choose $k = 5$ and perform 5-fold cross-validation to evaluate the linear regression procedure as correctly as possible on our limited dataset. We have at most 100 data points for a given property because, as pointed out before, we have a total of 100 element names and their corresponding embeddings. Furthermore, not every element in our dataset has a specified value for a given property. For example, as the mendeleev package uses the 18-column form of the periodic table, Ce-Lu and Th-Lr are not assigned a column number as they are placed between columns 3 and 4. Removing these elements leaves us with 76 elements for the predictions of the right column in the periodic table. In the case of Pauling electronegativity, the noble gasses generally do not bond with other atoms and thus an electronegativity can not be determined. Furthermore, the Pauling electronegativity of some rare elements is simply unknown as they are not well studied (yet). This leaves us with 85 elements for the property electronegativity. The latter reason also goes for the boiling point, evaporation heat, heat of formation and melting point. For these properties, we respectively have 95, 88, 87 and 98 elements left.

To interpret the quality of the linear regression procedure, we construct a plot of the predictions and the true values. Perfect predictions would yield a straight line ($y = x$). The R^2 value or the goodness-of-fit measure for linear regression models, is also calculated. As we know, R (a number between 1 and -1) represents the relationship between variables. R^2, ranging from 0 to 1, is used here to check the correlation between the predicted property values and the true values. An R^2 value of 1 would indicate a perfect correlation.

Let's now go over to the results of the linear regression procedure with the 15-dimensional embeddings and various atomic and thermodynamic properties.

2.2.2 Results

The results for various atomic properties can be seen in Figure 2.3.

On Figure 2.4, the results for the thermodynamic properties are displayed.

The true value is given on the horizontal axis, the predictions on the vertical axis. The black dashed line is included for clarity and indicates the true value on both axes. Each colour represents one validation set originating from the 5 folds that were created during the cross-validation procedure. The resulting coefficient of determination (R^2) of the true data points versus the predicted values, is also given. On each plot, five outliers with the highest absolute error for a given property are indicated with their respective element name.

2.2.3 Discussion

Overall, it seems that the predictions of the linear regression models are quite good, with R^2 values between 0.45 and 0.75. However, when we look at the results by Tshitoyan et al. in the supplementary information of the paper (Supplementary Figure 4), they obtain R^2 values between 0.70 and 0.80 for similar properties. This is mainly due to the fact that they seem to have removed certain elemental embeddings before performing the linear regression[3]. As they were not clear in which embeddings they removed, we decided to stick with all of the embeddings that were available for a certain property and draw conclusions from these results. Guided by the outlier data points, the following conclusions can be made.

[3]This can clearly be seen when we compare the graphs with the predictions for row in the periodic table. Elements in row 1 and 7 were removed from the dataset by Tshitoyan et al. The exclusion of elemental embeddings is also apparent on the graphs for atomic weight and Mendeleev number.

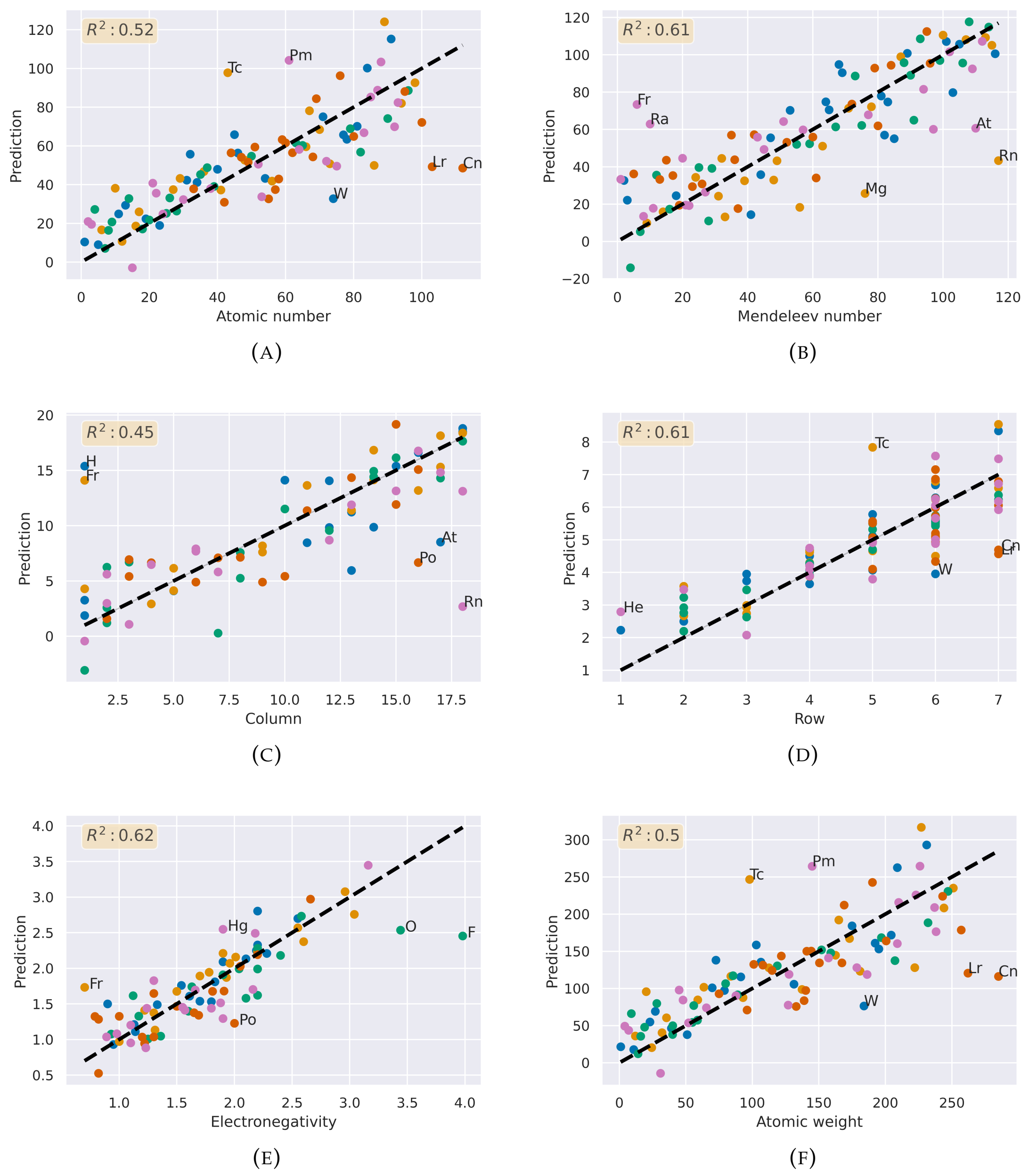
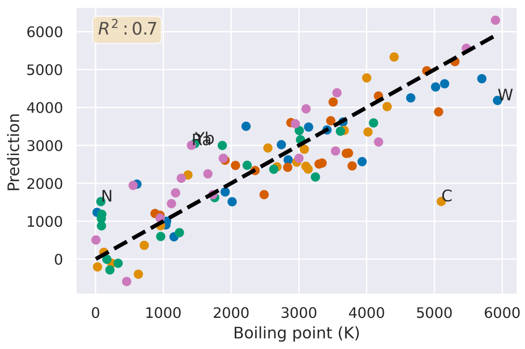
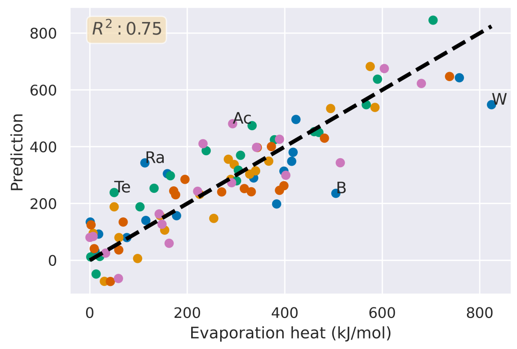


FIGURE 2.3: Linear regression with 5-fold cross-validation for atomic properties. The different colours refer to the five validation sets that were created in the 5-fold cross-validation procedure. The coefficient of determination, the R^2 value, is given on each plot. The black dashed line is included for clarity and indicates the true value on both axes. The outlier data points are indicated with their respective element name.

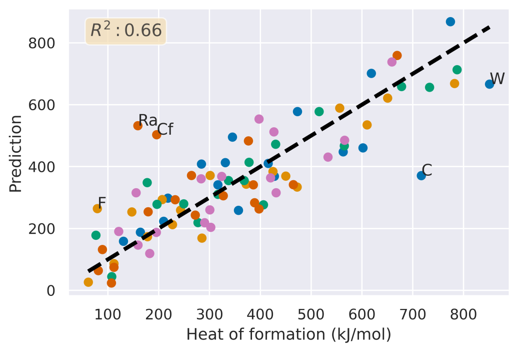
First of all, it immediately stands out that lawrencium (Lr), copernicium (Cn), californium (Cf), astatine (At), promethium (Pm) and francium (Fr) are marked quite frequently as an outlier data point on Figure 2.3. These elements are very rare and not well studied experimentally. For this reason they are not included in Figure 2.4 as there is no data available on their thermodynamic properties. It does not come as a surprise that these rare elements are marked at least twice as outliers for the atomic properties. Elements that are not well studied yet are also mentioned less frequently



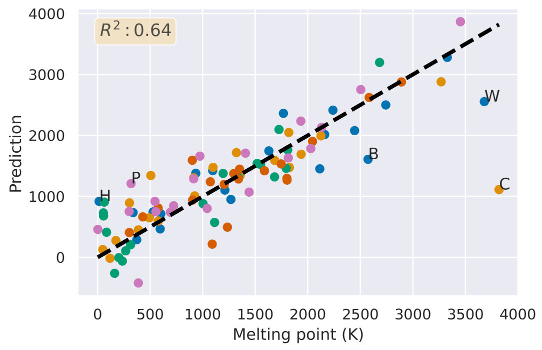
(A)



(B)



(C)



(D)

FIGURE 2.4: Linear regression with 5-fold cross-validation for thermodynamic properties. The different colours refer to the five validation sets that were created in the 5-fold cross-validation procedure. The coefficient of determination, the R^2 value, is given on each plot. The black dashed line is included for clarity and indicates the true value on both axes. The outlier data points are indicated with their respective element name.

in the materials science literature on which the model was trained. All together, we found that they appeared less than 0.005 % of the time when an element name was mentioned in the training corpus of Mat2vec. The quality of the embeddings reflects this. We can thus assign the bad performance of these uncommon elements to their poor embeddings.

Further visual inspection signals another problem with the embeddings or more specifically, the way in which they were retrieved from the Mat2vec model. Let's take tungsten (W) as an example, which is marked three times as outlier on Figure 2.3 and four times on Figure 2.4. The full name of this element in mendeleev is tungsten, so the embeddings of this element were retrieved by that name. However, another popular name for this element is wolfram. When we change its name from 'tungsten' to 'wolfram' and perform the linear regression, all R^2 values for the atomic properties are seen to improve and W stops being an outlier data point, except for Pauling electronegativity. The improvement on the R^2 values is less spectacular for the thermodynamic properties and even drops slightly for heat of formation. W is still an outlier for evaporation heat, heat of formation and melting point. It thus seems that the quality of the property predictions of an element is highly dependent on the element name that was used for retrieving the embedding.

The same reasoning can be made for carbon (C), which is marked three times as an outlier data point on Figure 2.4. C is the most common element name in the training corpus: it appears 10% of the time when an element name is mentioned. In contrast to tungsten, carbon is not known under any other name. However, when carbon is mentioned in materials science articles, it refers to different structures with very different properties. For example, 'carbon' could refer to diamond, graphite or carbon nanotubes. The same is true for other elements. E.g. boron (B), which is an outlier data point on Figures 2.4b and 2.4d, knows many applications when it forms a crystal compound with carbon. Boron carbide is an extremely hard material that is used in bulletproof vests, tank armor and various other industrial products[20]. This wide application span brings along many mentions in materials science articles, clouding the word vector for the true boron crystal.

A third issue is the bias that is present in the embeddings. It was mentioned in Section 2.1 that Radon (Rn), radium (Ra), technetium (Tc) and polonium (Po) seemed out of place in the two-dimensional embedding space. These radioactive elements were seen to appear closer to uranium (U) and thorium (Th) than to elements of their respective category. This bias in the literature is also reflected in the embeddings. When we look at Tc for example, it is marked as an outlier on Figure 2.3a, Figure 2.3d and Figure 2.3f. On each of these plots, the predicted value approaches that for U (respectively 92, 7 and $\approx$ 238) and Th (respectively 90, 7 and $\approx$ 232). On Figure 2.3e, the same is true for Po as U has an electronegativity of 1.7 and Th of 1.3. U and Th have no column value as mentioned earlier, so the comparison for Po and Rn to U and Th can't be made on Figure 2.3c. On Figure 2.3b, where both Rn and Ra appear as an outlier, the same logic as before can't be followed as the Mendeleev number of uranium is 20 and 16 for thorium.

Nonetheless, the bad prediction of the Mendeleev number of Rn brings up a fourth problem. Rn has the second to highest Mendeleev number and is greatly underestimated by the linear regression model. The same trend can be noticed on Figure 2.4, where the extreme thermodynamic properties of W are always underestimated by the linear regression model. Also boron (B) and carbon (C) are greatly underestimated. The same is true for oxygen (O) and fluorine (F) on Figure 2.3e, which are the elements with the highest Pauling electronegativity (respectively 3.44 and 3.98). It thus seems that the linear regression model fails to predict extreme values. The explanation behind this phenomenon is the following. When these extreme numbers are not part of the training data during the 5-fold cross-validation, the resulting linear regression model is more likely to make lower predictions when it comes to predicting the property values.

2.2.4 Mean R^2 value and standard deviation for multiple iterations

In Section 2.2.2, we performed 5-fold cross-validation once and plotted the results and the R^2 value. In their paper, Tshitoyan et al. have also calculated the mean R^2 value and its standard deviation for multiple iterations. As we also experienced, the cross-validation procedure converges quite fast because it samples homogeneously, so a different approach was used. During each run, random splitting of the dataset into a training (80%) and validation set (20%) was performed. The result of 20 randomly chosen train/validation sets with the average R^2 value and standard deviation of the validation set is depicted in Table 2.1, along with the 5-fold cross-validation results that were obtained earlier. The results by Tshitoyan et al. are also given for the properties that they investigated. Notice that more properties were included in this work. As mentioned earlier, the better results from Tshitoyan et al. seem to originate from the fact that they have removed certain elemental embeddings before performing the linear regression procedure.

Property	**This work**		**Tshitoyan et al.**	
	R^2	**avg. R^2 (20 runs)**	**R^2**	**avg. R^2 (20 runs)**
Atomic number	0.52	0.62±0.03	/	/
Mendeleev number	0.61	0.66±0.04	0.73	0.74±0.09
Column	0.45	0.25± 0.15	0.71	0.69±0.16
Row	0.61	0.65± 0.03	0.80	0.81±0.07
Electronegativity	0.62	0.71±0.03	0.75	0.76±0.12
Atomic weight	0.50	0.61±0.04	0.75	0.72±0.10
Boiling point	0.70	0.77±0.02	/	/
Evaporation heat	0.75	0.80±0.02	/	/
Heat of formation	0.66	0.67±0.03	/	/
Melting point	0.64	0.71±0.03	0.75	0.74±0.10

TABLE 2.1: Average R^2 value with standard deviation of the validation set for atomic and thermodynamic properties after 20 runs. During each run, the dataset was randomly split into a training set (80%) and a validation set (20%). For completeness, the 5-fold cross-validation results from earlier are included in the table. The results from Tshitoyan et al. are also shown.

Except for the properties column (slightly lower within the standard deviation) and heat of formation (equal within the standard deviation) the average R^2 values are better compared to those in Section 2.2.2. The reason for this is that the cross-validation procedure ensures that every data point is used at least once in the validation set, which is not the case in the random splitting approach. More frequent inclusion of certain data points in the training set might yield other, in this case better, results. Elements with qualitative embeddings in the training set allow for a better linear regression model.

2.2.5 Finding the optimal number of PCA components

The standard deviations on the average R^2 values in Section 2.2.4 are reasonable, except for the column property. As mentioned before (Section 2.2.1) we only have 76 elements with a column value. As can be expected with a low R^2 value, the root mean square error (RMSE) on the predictions in the validation set is quite high given there are only 18 columns: 4.24 ± 0.28. It seems that we are in overfitting territory. Let's try to reduce the dimensionality of the 15-dimensional embeddings even more to see if that yields better results.

After varying the number of PCA components and calculating the average R^2 value with standard deviation after 20 runs, we find that 7 PCA components give the best results. The explained variance ratio is lower (46% compared to 65% for 15 PCA components), but $R^2 = 0.57 \pm 0.04$ and the RMSE equals 3.4 ± 0.16. The R^2 value is significantly better and its error is much lower. The RMSE is still high, but lower than it was before. Altering the number of PCA components thus proved to be successful to optimize the performance of the linear regression model.

Inspired by this result, the influence of the number of PCA components on the performance of the linear regression model was also investigated for the other atomic and thermodynamic properties. However, no smaller or larger number of PCA components yielded a better R^2 value within the standard error. We can thus conclude that dimensional reduction to 15 PCA components is justified for all properties, except for the column value of the periodic table where reduction to 7 PCA components seemed best.

2.3 Conclusion

As we saw, R^2 values of the properties in Figures 2.3 and 2.4 are good, but they could be better. The hypothesis that the dimensional reduction with PCA was somewhat responsible for the varying results, was proven false in Section 2.2.5, except for the column property.

Guided by the outliers on Figures 2.3 and 2.4, there seemed to be four main problems: poor embeddings of rare elements, ambiguous element names, bias in the embeddings and the struggle of the linear regression model to predict extreme values. Possible solutions to these problems are discussed below.

1. Rare elements such as lawrencium (Lr), copernicium (Cn), californium (Cf), astatine (At), promethium (Pm) and francium (Fr), have poor embeddings because they are almost never mentioned in the literature. We could try to remove them from the dataset, but then we would end up with even less data points.

2. The problem that 'W', 'tungsten' and 'wolfram' all yield different embeddings, could be solved by normalizing the embeddings, such that unambiguous retrieval of the embeddings becomes possible. However, the problem remains in the example case of carbon that was given. There it would be better to split up the embeddings, depending on the context. For example, separate embeddings for diamond and graphite could be constructed, but this approach would obviously require some effort.

3. The bias in the embeddings of radioactive elements is impossible to get rid of because the very reason for this bias is the nature of the embeddings. Radioactive elements are likely to appear in similar contexts and this is reflected in the embeddings.

4. The struggle to predict extreme values is due to the limited dataset as only a relatively small amount of elements exist in the universe, but also due to the simple linear model that we are using here. It is highly assumptious that linear functions suffice to connect word embeddings to specific properties. More complex non-linear models should be investigated. However, one must keep in mind that fitting more parameters with a limited dataset can lead to overfitting.

In contrast to the limited number of elements in the periodic table, we can combine them into crystal structures with multiple elements in an infinite number of combinations. This is done in the next chapter to predict the formation energy of quaternary crystal structures. A major difference with this chapter is that we will no longer use a linear regression model to make the predictions. Instead, we will turn to methods that can identify non-linear relationships: random forests and neural networks.

Chapter 3

Predicting the formation energy of ABC_2D_6 elpasolites

In Chapter 2, we looked at elemental embeddings to investigate atomic properties and thermodynamic properties of crystals made up of a single element. Let's now look at crystals containing multiple elements and see if their combined embeddings can be of use to predict a specific property. More specifically, we will concatenate the embeddings of elemental crystals and treat the result as the embedding of a crystal.

Tshitoyan et al. have recently shown[1] that embeddings can be of use in machine learning models, such as for predictions of the formation energy of crystals. The formation energy is defined as the difference in energy between a crystal and corresponding amounts of unary crystals for every element that appears in the more complex crystal. A material is said to be thermodynamically stable if the formation energy is negative, meaning that energy is released when forming the material[1]. Furthermore, the formation energy is a useful quantity for the calculation of reaction enthalpies, the generation of phase diagrams and the determination of many other material properties[21].

The ubiquity of the formation energy makes it an interesting property to calculate for materials science purposes. One can calculate the formation energy, and lots of other properties, with density functional theory (DFT). The idea behind DFT is that the electron density and the total energy of a system uniquely define each other. Since the 1970s, DFT has been the most successful ab initio method to solve problems in chemistry and materials science. The great advantage of DFT is that it avoids solving the Schrödinger equation directly, which is very hard to solve analytically. Instead, the DFT Kohn-Sham equations are a set of differential equations that reduce the many-body problem to a number of easier to solve one-body problems. However, solving differential equations does take a significant amount of time, especially if we want to do these calculations multiple times for different compounds.

The dawn of machine learning in materials science has given us a way to bypass the Kohn-Sham equations from DFT, making it possible to predict the formation energy in milliseconds. This is what Faber et al.[22] have done for the calculation of the formation energy of elpasolites. Elpasolites are quaternary crystal structures with the general formula ABC_2D_6. As four different elements make up one elpasolite, millions of combinations are possible. Faber et al. constructed a machine learning model for the calculation of the formation energy of elpasolites, hereby reaching

[1]However, this does not imply that the material is truly stable as the same chemical composition can possibly have a more negative formation energy for other crystal structures.

similar accuracy as DFT methods. We will use the precalculated formation energies from their machine learning model to train our own models and to validate the results. We thus will not start from DFT information. Instead, we will rely on the dataset generated with machine learning from Faber et al. Before taking a closer look at the dataset, we first introduce elpasolites.

3.1 Elpasolites

Elpasolites, with the general formula ABC_2D_6, are the predominant quaternary crystal structure in the Inorganic Crystal Structure Database (ICSD). They are of special interest due to their scintillating properties, i.e. they emit light when excited by ionizing radiation. More specifically, they re-emit (a fraction of) the energy that was absorbed from the incoming particle in the form of light. This makes them ideal as scintillator devices, which are used in experimental particle physics to detect ionizing particles such as electrons, alpha particles or high-energy photons. They are also commonly used for security purposes (e.g. cargo scanning with X-rays) and in medical facilities (CT scanners). The structure of an elpasolite is depicted in Figure 3.1 by means of 4 unit cells. The positions of the A, B, C and D atoms are indicated on the figure. One can verify that the unit cell, which is the smallest repeating unit, indeed contains one A atom, one B atom, two C atoms and six D atoms.

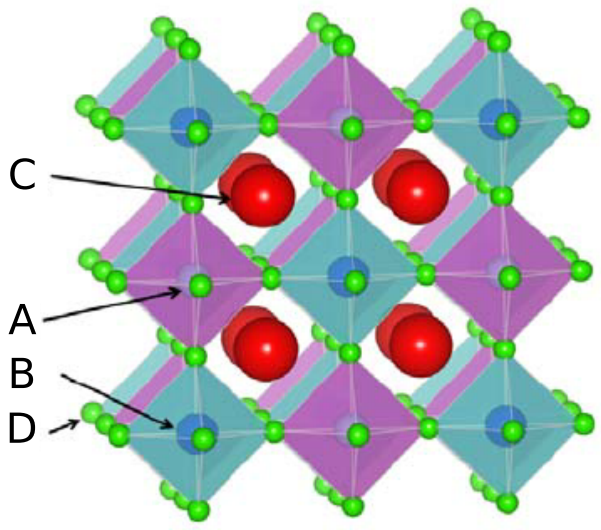


FIGURE 3.1: Structure of an elpasolite (ABC_2D_6)[23]. The positions of the A, B, C and D atoms are indicated.

Furthermore, the Crystallography Open Database (COD)[24] tells us that the space group of elpasolites is fm-3m (number 225) and its symmetry class is cubic. The prototype of an elpasolite is $AlNaK_2F_6$. The Wyckoff positions are: 4a for Al, 4b for Na, 8c or K and 24e for F. This last position has one degree of freedom.

3.2 The dataset by Faber et al.

The dataset that is used in this chapter was constructed by Faber et al. Using machine learning, they calculated the formation energy of all the elpasolites that can be formed from main-group elements up to bismuth (Bi). These 39 elements yield a total of $39 \cdot 38 \cdot 37 \cdot 36 = 1,974,024 \approx 2 \cdot 10^6$ different elpasolites, consisting of four different elements on the A, B, C and D positions. With a training set of $1 \cdot 10^4$ crystals for which they generated the DFT formation energy, they obtained a mean absolute error of ± 0.1 eV/atom, which is comparable to the DFT accuracy for solids.

For our purposes, the complete dataset from Faber et al. with the machine learning generated formation energy of 1,974,024 elpasolites was split into a training set (80%), a validation set (10%) and a test set (10%). The number of samples in each subset can be seen in Table 3.1.

Type	**Samples**
Training set	1,579,220
Validation set	197,402
Test set	197,402
Total dataset	1,974,024

TABLE 3.1: Overview of the number of samples in the training, validation and test set after performing the 80-10-10 split.

A histogram with the distribution of the formation energy in the dataset is given in Figure 3.2. As stated before, a material is thermodynamically stable if the formation

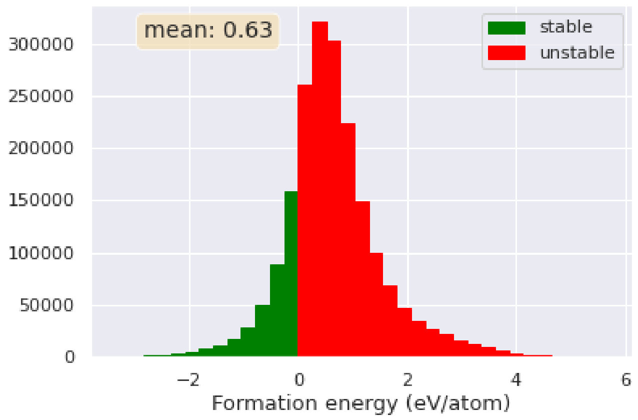


FIGURE 3.2: The distribution of the formation energy in the dataset. Stable materials have a negative formation energy (green). For a positive formation energy, the material is predicted to be unstable (red).

energy is negative (green) and unstable if the formation energy is positive (red)[2]. Given the abundance of red bins on Figure 3.2, most elpasolites in the dataset are unstable. This is also reflected by a mean formation energy of 0.63 eV/atom in the dataset. Thermodynamically unstable materials do not occur in nature as energy is needed during the formation of these materials.

[2]Often a tolerance of 0.05 eV/atom is allowed for DFT error and metastable materials.

3.3 Random forest

We will first turn to a random forest to predict the formation energy of the elpasolites. As we have the precalculated formation energies from Faber et al. at our disposal, we can create a random forest to predict the formation energy of the elpasolites given the atomic numbers of the A, B, C and D element. We use the random forest regressor from scikit-learn[18] to create the random forest. The random forest consists of multiple decision trees. As the target variable is a continuous value, the decision trees in the random forest are also called regression trees.

How can a regression tree be used to predict the formation energy? An example of a single regression tree is given in Figure 3.3. In the top node, all data is in one group

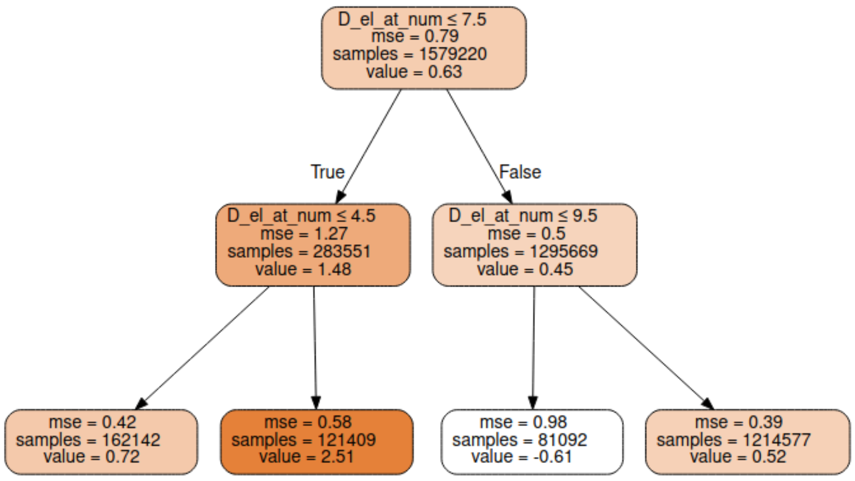


FIGURE 3.3: An example of a single regression tree with scikit-learn.

and the best prediction for the formation energy is simply the average formation energy of all elpasolites in the dataset: 0.63 eV/atom. Subsequently, the decision tree will impose decision criteria that maximize the homogeneity of the data points in the resulting sub-nodes. To decide which decision criterion will be most efficient, the decision tree will split the nodes on all available input variables and then select the split which results in the most homogeneous sub-nodes. For our particular case, the decision criterion determined by the model for the first split is that the atomic number of the D element is equal to or less than 7.5. All the samples that meet this criterion go to one internal node, those that don't to the other internal node. This process with suggested decision criteria from the model repeats itself until it reaches a stopping criterion. The stopping criterion in this example was a maximum of four leaf nodes, which are nodes without any further splits.

To make the predictions for the formation energy, special attention was paid to the following parameters.

min_samples_leaf and min_samples_split

Specifying when to stop the splitting of the tree nodes is important. With scikit-learn's default settings, the splitting of nodes would continue until there is only one sample in each leaf node. This causes the model to overfit (i.e. it fails to generalize its predictions), meaning that the model does not perform well on data that it hasn't seen before. As a consequence, the predictions of the formation energy in the validation set and test set would come with large errors.

Limiting the depth of the tree to avoid overfitting based on the number of samples can be done with the following parameters in scikit-learn. The parameter *min_samples_leaf* guarantees a minimum number of samples in a leaf node and *min_samples_split* is the minimum number of samples required to split an internal node. An example clarifies the practical implementation of these parameters. Suppose we have *min_samples_split*$=10$, *min_samples_leaf*$=5$ and 15 samples arrive at an internal node. The split is allowed because $15 \geq 10$. Let's say that it results in two leaves: one with 4 samples and one with 11 samples. However, as we end up with 4 samples in one internal node and *min_samples_leaf*$=5$, the split won't be allowed because the resulting leaf with 4 samples does not have the minimum number of samples required to be at a leaf node.

n_estimators

Another important parameter is *n_estimators*, which defines the number of trees in the random forest. As the random forest averages the predictions of all trees in the forest, it is important to have plenty of trees. The default value of *n_estimators* is 100, but how can we tell if this is enough? We can investigate this by obtaining the predictions of each individual tree and watch what happens to the root-mean-square error (RMSE) of the predictions in the validation set as more trees are added. The RMSE as a function of the number of trees in the forest is plotted in Figure 3.4. At

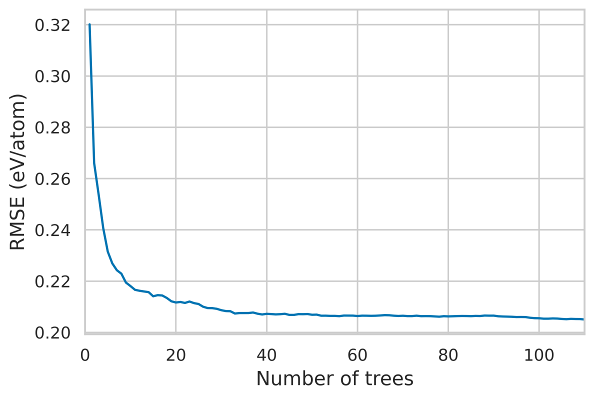
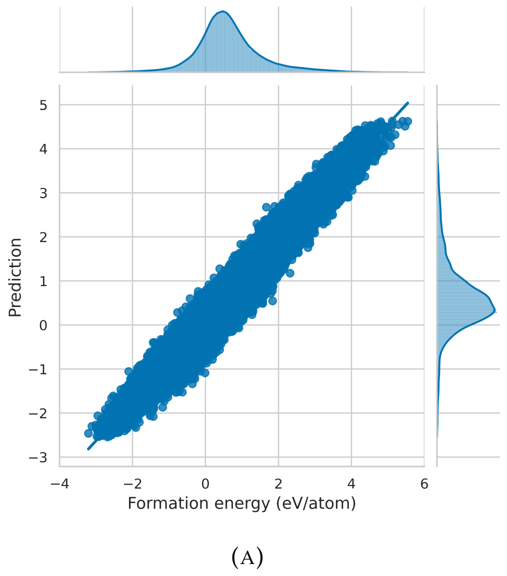


FIGURE 3.4: The RMSE as a function of the number of trees in the random forest.

around 100 trees, the RMSE is seen to stagnate. The default value of 100 trees thus seems fit for our purposes.

3.3.1 Results

The results of the formation energy predictions on the test set are plotted in Figure 3.5. They are obtained with *n_estimators*=100, *min_samples_split*$=5$ and *min_samples_leaf*$=5$. The results for the training and validation set are included in Appendix A.1. Figure 3.5a shows the predictions of the formation energy in function of the true formation energy, along with a linear regression fit. On the outer vertical



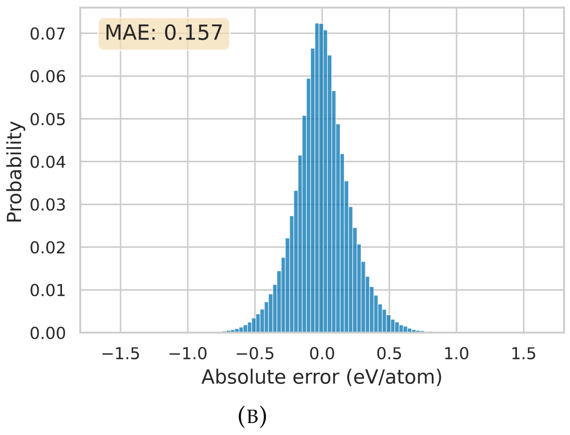


FIGURE 3.5: The results of the formation energy predictions on the test set with a random forest.

		Predicted	
		Stable	Unstable
Actual	Stable	0.750	0.250
	Unstable	0.013	0.987

TABLE 3.2: Normalized confusion matrix for the classification of stable and unstable elpasolites in the validation set after 25 epochs, with a random forest.

and horizontal axis, a histogram[3] that indicates the data distribution of the opposing axis is shown. The error histogram on Figure 3.5b shows the distribution of the absolute errors on the predictions, i.e. the true formation energy minus the predicted formation energy. The mean absolute error (MAE), which is the mean value of the difference between the true and the predicted formation energy in absolute value, is also shown on the plot.

We see that the MAE is 0.157 eV/atom, meaning that the predictions of the formation energy were over- or underestimated by 0.157 eV/atom on average. What does this mean in practice for the prediction of stable and unstable materials? As we remember from Figure 3.2, most elpasolites in our dataset are unstable. They are characterized by a positive formation energy value. The stable elpasolites have a negative formation energy. After the predictions of the formation energy by the random forest, the materials can again be classified as stable or unstable, but now based on the predicted value of the formation energy. The confusion matrix of the classification into stable and unstable elpasolites is shown in Table 3.2.

The diagonal represents the normalized percentage of correct predictions for the stable and unstable materials. The incorrect predictions are represented by the

[3]More specifically, a kernel density estimate (KDE) plot is shown here. KDE visualizes the distribution of the data with a continuous probability density curve in one or more dimensions.

off-diagonal elements. Almost all of the unstable materials were correctly predicted as unstable, but 25% of the stable materials were incorrectly classified as unstable, based on the prediction of the formation energy. As most of the formation energies in the dataset are positive, the random forest is more likely to predict a stable elpasolite as unstable. Hence, we must turn to another approach.

Tshitoyan et al. have recently shown[1] that embeddings can be of use in machine learning models. Inspired by this, we will apply deep learning and neural networks to get better predictions for the stable elpasolites, hereby including embeddings. Before constructing our own neural network with embeddings, we will first introduce the basic concepts of deep learning.

3.4 The basics of deep learning and neural networks

As stated by Goodfellow et al. in their book 'Deep Learning'[25], deep learning is a specific kind of machine learning. The notion 'deep' indicates that the input data travels a longer path before it reaches the output of a deep learning model, as compared to models in machine learning. More specifically, we say that multiple *layers* are present between the input and output of the model.

With deep learning, one can construct a deep neural network. A deep neural network tries to mimic the brain functionality of a human being with artificial neurons. There are different types of neural networks, but they always consist of the following components: *neurons*, *synapses*, *weights*, *biases*, and *activation functions*.

Much like biological neurons, artificial *neurons* are the basic units of neural networks that receive information, perform calculations and then pass it on. Three classes of neurons can be distinguished: input neurons (receive information), hidden neurons (process information) and output neurons (produce conclusion). Neurons are organized in *layers*. Input neurons make up the input layer, hidden neurons come from the hidden layers and the output neurons are situated in the output layer. As opposed to the input and output layer, multiple hidden layers can exist in the deep neural network. An illustration of the neurons and the layers in a neural network is given in Figure 3.6.

The neurons are connected with one another by *synapses*. Every synapse has a *weight*. As the name 'weight' suggests, this is a way to lower or raise the effect of a particular neuron on the neuron with which it is connected through the synapse. The results of neurons with high weights will be dominant, while neurons with low weights will be granted little influence. In the beginning, all of the weights are randomly initialized. They are optimized during the training process of the deep neural network. Apart from weights to steer the processing of information, a *bias* is added to every layer. A bias is a constant value (i.e. not influenced by the previous layer) and can be compared to the intercept of a linear equation. Just like the weights, the value of the bias is optimized during the training process. We thus get the following computation in a neuron j for input values x_i and synapses with weights W_{ij}

$$y_j = f(\sum_{i=1} W_{ij} \cdot x_i + b), \tag{3.1}$$

with y_j the output of the neuron and b the bias term. The function f is the *activation function*, which maps its argument to a certain range before being sent as output to the next layer. The activation function that is used in this chapter is the Rectified

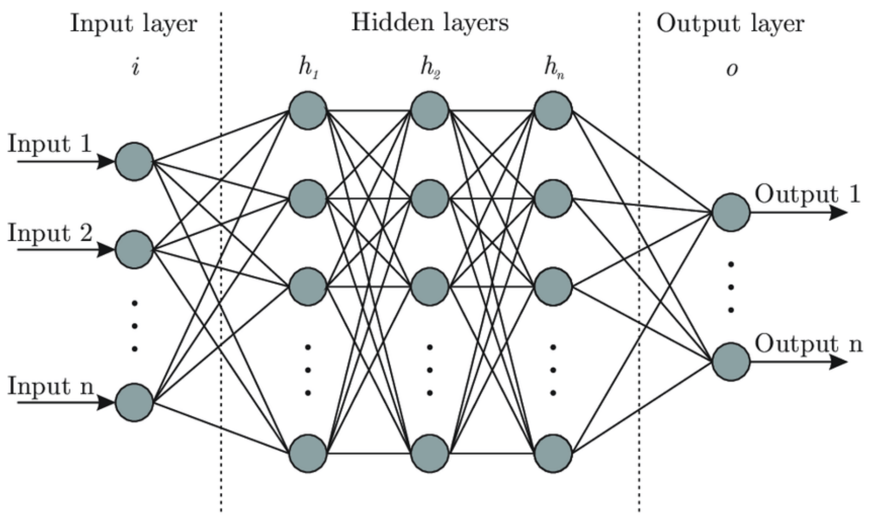


FIGURE 3.6: An illustration of the neurons and layers in a neural network[26]. The input layer contains input neurons, the neurons in the hidden layers are hidden neurons and output neurons are situated in the output layer. These neurons respectively receive information, process information and produce conclusions.

Linear Unit (ReLU) function:

$$f(x) = \max(0, x) = \begin{cases} x_i, & \text{if } x_i \geq 0 \\ 0, & \text{if } x_i < 0 \end{cases} . \tag{3.2}$$

Another popular activation function is the sigmoid function, which has a characteristic S-shaped form. A common example of a sigmoid function is the logistic function:

$$S(x) = \frac{1}{1 + e^{-x}}. \tag{3.3}$$

However, in our neural network, it will not be implemented as an activation function. Instead, the sigmoid function will be used to translate the predictions of the formation energy to a number in between the range of the formation energy in our dataset, which is approximately $[-3.5, 6]$ eV/atom as can be seen on Figure 3.2.

Now that we know the general components of a neural network, we will also go over some relevant parameters and concepts that are of special importance during the training process of a neural network.

Epoch

When all samples in the training dataset have passed through the neural network once, we say that one epoch has passed. The data is not fed to the model all at once, but is grouped together in batches. The neural network will update its performance after one batch has passed, which is also called an iteration. For example, if we have 1000 samples and a batch size of 100, 10 iterations or updates will occur in one epoch. To allow the neural network to learn, we will pass the entire dataset multiple times through the neural network, meaning that multiple epochs are performed.

Loss

What does it mean for a model to update its performance? To understand this, we must introduce the most important concept when training a neural network: the loss. Before we can understand the loss, we must first look at the loss function i.e. the function that is used to calculate the loss. The loss function is important because it allows us to evaluate the model's weights that are present in every layer of the neural network.

The loss function that is used in this chapter is the mean square error (MSE) loss function, which is most commonly used in regression problems. In an ideal world, the MSE loss with optimal weights would be zero, which means that all predictions are exactly equal to the true values. In practice, we want to make the loss as low as possible by gradually optimizing the weights. This is a high dimensional optimization problem given that we have lots of weights.

During the training process of a neural network, the weights are optimized with gradient descent. The idea behind gradient descent is as follows. After every iteration, the gradient of the loss function is calculated. The gradient is a vector that contains all of the partial derivatives of the loss function. The partial derivative is calculated by derivation of the loss function with respect to one weight, while keeping the other weights constant. As we know from high school, the derivative of a function tells us the slope of that function. Given that we are looking for the minimum loss, a very large slope (in absolute value) suggests that we are still far removed from this objective, while a small slope (in absolute value) hints that we are almost there. The sign of the derivative tells us which way to go. These basic principles are used to find the minimum loss in gradient descent.

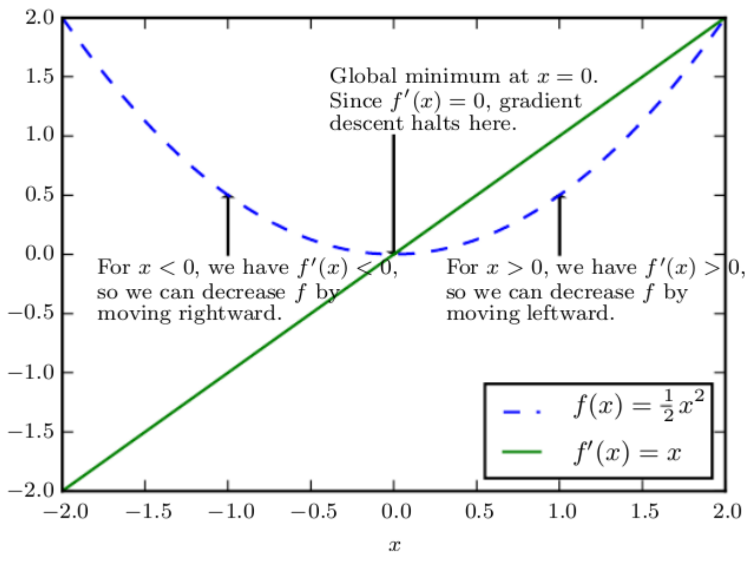


FIGURE 3.7: Using gradient descent to find the minimum of a function. The derivative (green) is largest in absolute value the further we are from the minimum of the function (blue). Taking the sign of the derivative into account, we must simply step down the curve until the derivative flips sign. When it does, we know we have crossed the minimum of the function[25].

A simplified example of gradient descent with one variable from the book 'Deep Learning' by Goodfellow et al.[25] is given in Figure 3.7. On the figure, the function

$\frac{1}{2}x^2$ (blue) and its derivative (green) are plotted. The minimum of the function is at $x = 0$, where also the derivative has its minimum value. The further away we are from the minimum, the greater in absolute value the derivative becomes. When the sign of the derivative is negative, we must move to the right to find the minimum. When it is positive, we must go left. Stepping down the curve until the derivative changes sign, will give us the minimum of the function.

The neural network will perform a similar analysis (but now with a lot more variables) to decide on how to change the weights. The weights are changed with a so-called step size. The step size is determined by multiplying the gradient by a number of our own choosing: the learning rate.

Learning rate

The learning rate is a parameter used for the gradient descent. Choosing the learning rate is important because a learning rate that is too small will require the model to take a lot of steps before the minimum is reached. Also, we must be careful not to get stuck in a local minimum, which can happen if the step size is too small. With a learning rate that is too high, we risk overshooting the minimum with a diverging loss as result. To help us with this decision, the learning rate finder can be used. The learning rate finder plots the learning rate in function of the loss. An example is plotted in Figure 3.8. After some initial fluctuations we can see that the model's loss stays more or less constant, i.e. the model does not train, in the range 10^{-6} to 10^{-4}. Then the loss starts decreasing and eventually it reaches a minimum after which the loss diverges. A rule of thumb for picking a learning rate is to choose a value that lies comfortably before the minimum, but still in the range where the loss was clearly decreasing. For this particular example, a learning rate around 10^{-3} would be appropriate.

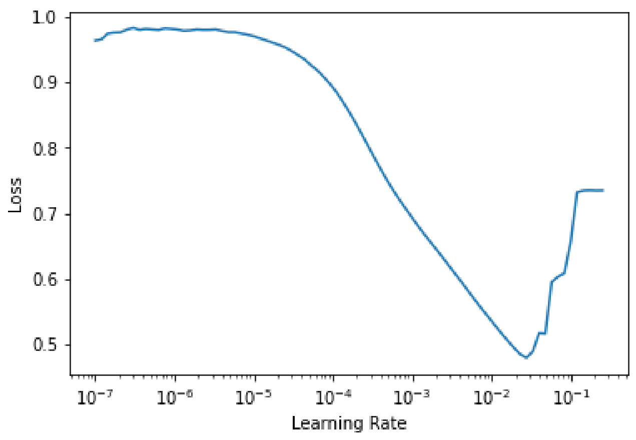


FIGURE 3.8: An example of the learning rate finder, which plots the learning rate in function of the loss.

3.5 Neural network with Mat2vec embeddings

After the theoretic foundation of the previous section, it is now time to turn this into practice. Fastai[27] and PyTorch[28] will be used to construct and train the neural networks. We will use the Mat2vec embeddings of the constituent elpasolite atoms as weights for the neural network in the input layer. In contrast to Chapter 1, the embeddings of the elements are retrieved by their respective element symbol (e.g. 'H'). We now put these embeddings in a tensor, which can be described as a collection of numbers in a particular shape. As there are 118 elements in total and each element has an embedding vector of 200 dimensions, we create a PyTorch tensor of shape 118×200. It would be convenient to identify the embedding of an element by its atomic number, so we add a row of zeros to the tensor. The final tensor thus has a shape of 119×200 and allows for the retrieval of the correct embedding with the atomic number of an element. This tensor will be called the embedding tensor.

A schematic representation of the neural network that was constructed is shown in Figure 3.9. The neural network will take the atomic number of the A, B, C and

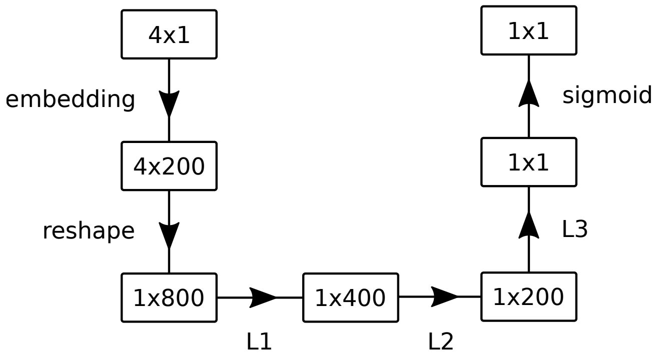


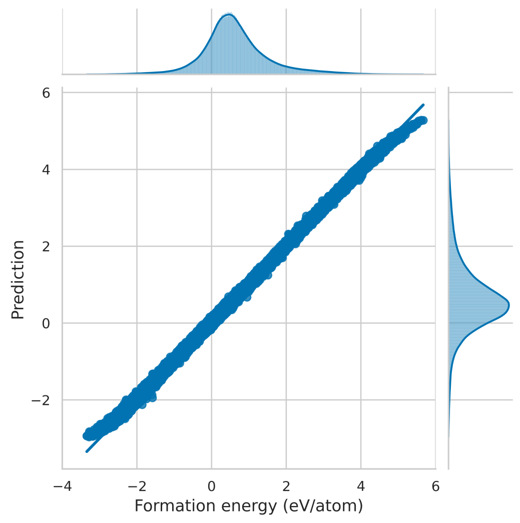
FIGURE 3.9: Schematic representation of the neural network architecture.

D elements as input (a 4×1 tensor). Each atomic number will then be converted to its corresponding embedding of 200 dimensions through the embedding tensor which gives us a tensor of 4×200. This tensor thus contains the initial weights of the neural network for a given elpasolite. After reshaping this tensor (1×800) two hidden layers L1 and L2 will be traversed, with 400 and 200 neurons respectively. The weights in these layers are optimized during the training process of the neural network. Finally, the last layer reduces the 1×200 tensor to a single output. The sigmoid function at the end makes sure the predicted value lies in between the range $[-3.5, 6]$, which is the range of the formation energy (in eV/atom) in our dataset.

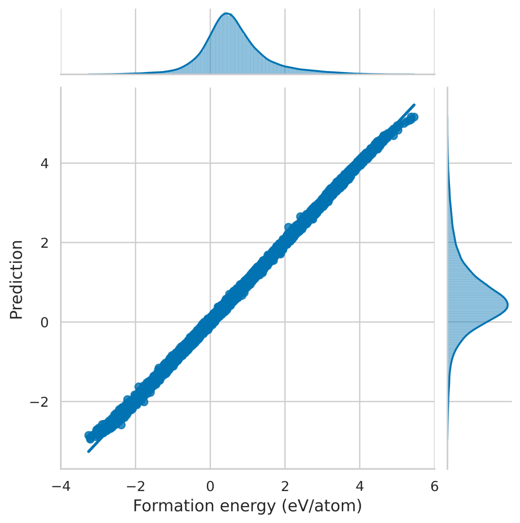
3.5.1 The result after 25 epochs

The results after training for 25 epochs with a learning rate of $1 \cdot 10^{-3}$ can be seen on Figure 3.10 for the training set and validation set.

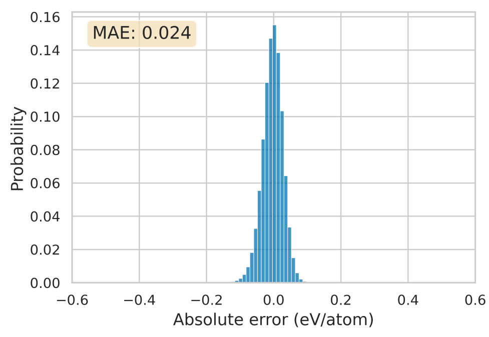
We immediately see that the results have improved when we compare them with the results of the random forest. The distribution of the data points on Figures 3.10a and 3.10b is less spread out. The predictions do deviate from the linear fit at the edges (which are data points that originate from the tails of the data distribution), but this can be explained by the rareness of these extreme formation energy values. The MAE of 0.024 eV/atom on Figures 3.10c and 3.10d is significantly better than for the random forest. The fact that we have the same MAE (up to 3 decimal places) for both the training and validation set implies that the model does not overfit. When overfitting occurs, the model stops finding generalizable patterns in the data and instead starts to memorize the data in the training set. This would lead to very good



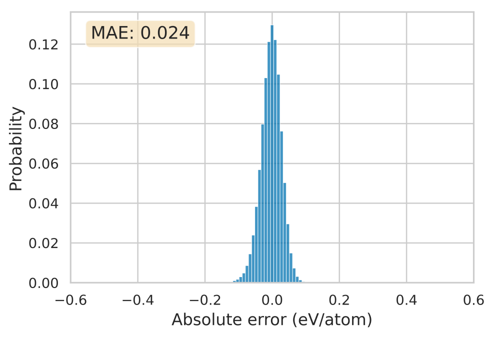
(A) Training set



(B) Validation set



(C) Training set



(D) Validation set

FIGURE 3.10: The results of the training set and validation set for the prediction of the formation energy after training the neural network with Mat2vec embeddings for 25 epochs.

results on the data in the training set, but poor performance on the validation (and test) set. This is luckily not the case here.

The confusion matrix of the validation set can be seen in Table 3.3. With only 2.08 % of the stable elpasolites and 0.62 % of the unstable elpasolites wrongly classified, we obtain a very good result.

		Predicted	
		Stable	Unstable
Actual	Stable	97.89 %	2.11 %
	Unstable	0.63 %	99.37 %

TABLE 3.3: Normalized confusion matrix for the classification of stable and unstable elpasolites in the validation set after 25 epochs, with a neural network and Mat2vec embeddings.

However, the training process does not have to stop already. We can continue training the model to see if we can achieve even better results.

3.5.2 Final result

During the training process, it is important to plot the learning rate finder frequently such that the learning rate can be altered in time. We performed the optimization process several times and found that we always ended up at the same results. One possible training procedure is shown in Table 3.4. This procedure is not set in stone, a different number of epochs or learning rates could also be used to reach the best possible result.

Epochs	Learning rate
10	$10 \cdot 10^{-4}$
5	$10 \cdot 10^{-6}$
5	$10 \cdot 10^{-7}$
5	$10 \cdot 10^{-7}$

TABLE 3.4: Possible training procedure for the neural network to obtain optimal results.

The final results for the training and validation set are included in Appendix A.2. Note that these results do not differ much from the results we obtained earlier after 25 epochs, meaning that the neural network was already trained well. A histogram with the absolute errors and the MAE of the predictions on the test set is shown in Figure 3.11. With a MAE of only 0.021 eV/atom, we were able to improve the model slightly.

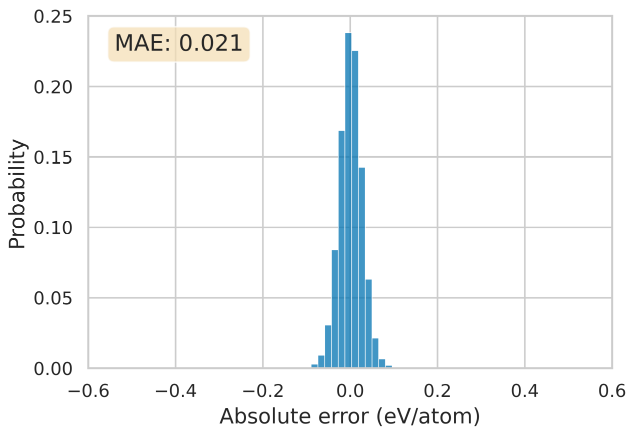


FIGURE 3.11: The final result of the prediction of the formation energy of elpasolites in the test set with a neural network and Mat2vec embeddings. The MAE is also given.

The confusion matrix has also improved after fully training the model. In Table 3.5, we can see that false classification of stable elpasolites was brought down to 1,93 % and less than 0.5 % of the unstable elpasolites were classified as stable.

At this point, we can conclude that our neural network with Mat2vec embeddings is fairly good at predicting the formation energy and labeling stable and unstable elpasolites in the dataset provided by Faber et al. The next question we will examine

		Predicted	
		Stable	Unstable
Actual	Stable	98.07 %	1.93 %
	Unstable	0.47 %	99.53 %

TABLE 3.5: Normalized confusion matrix for the classification of stable and unstable elpasolites in the test set, with a neural network and Mat2vec embeddings.

is whether this is due to the neural network architecture itself, due to the Mat2vec embeddings, or due to a combination of both.

3.5.3 Neural network with random embeddings

To examine this question, we constructed a neural network and replaced the values in the embedding tensor with random numbers. The final result for the model with random embeddings on the test set is shown in Figure 3.12. The results for the training and validation set are included in Appendix A.3.

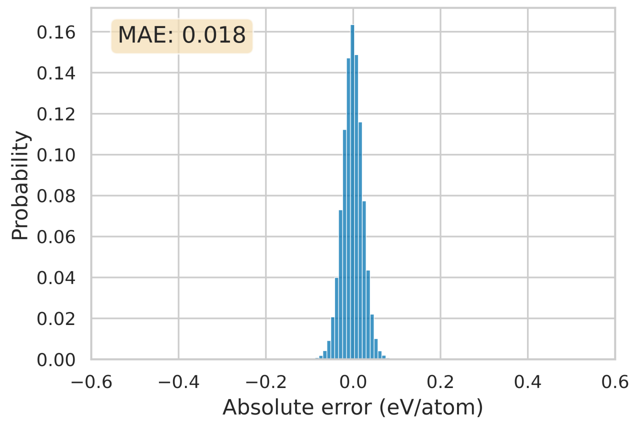


FIGURE 3.12: The final result of the prediction of the formation energy of elpasolites in the test set with a neural network and random embeddings. The MAE is also given.

As we can see, the overall predictions for the formation energy are slightly better with random embeddings as we obtain a mean absolute error of 0.018 eV/atom with the final model on the test set. However, this smaller error does not result in major differences in the confusion matrix, given in Table 3.6. Notice that the model with the Mat2vec embeddings performs slightly better when it comes to classifying the stable elpasolites, but the model with random embeddings does a better job at classifying the unstable elpasolites. Overall, the differences between the model with Mat2vec embeddings and the random embeddings are negligible. The use of Mat2vec embeddings thus does not seem an asset, as the embedding tensor resulting from a random initialization yields similar results.

		Predicted	
		Stable	Unstable
Actual	Stable	97.87 %	2.13 %
	Unstable	0.38 %	99.62 %

TABLE 3.6: Normalized confusion matrix for the classification of stable and unstable elpasolites in the test set after 25 epochs, with a neural network and random embeddings.

3.5.4 Discussion

We thus find that the Mat2vec embeddings do not improve the predictions of the formation energy of elpasolites. Maybe our large dataset is the reason for this? After all, one can expect that both methods will converge given there are millions of data points to extract relevant information from and quite some epochs to optimize performance. However, when testing with only a fraction of the dataset and less epochs, similar results were obtained.

We also tried to construct the embeddings of the elpasolites in a different way. Instead of concatenating the embeddings of the constituent elements, we averaged the 1×200 embeddings of the A, B, C and D elements into one embedding of 1×200 for the elpasolites. The number of weights in the subsequent layers was also lowered as the input layer now consisted of only 200 weights. Unfortunately, this worsened the performance of the neural network.

One last effort was done by fine-tuning the model after unfreezing the embeddings, i.e. also training the input embeddings instead of only the linear layers. In other words, we first trained the model with the embeddings kept constant and then we continued training the model whilst allowing the input embeddings to be altered. However, we did not obtain a better result.

3.6 Conclusion

In this chapter, our aim was to predict the formation energy of elpasolites, which are quaternary materials with the general formula ABC_2D_6. The dataset that was used for the predictions contained $2 \cdot 10^6$ elpasolites with the corresponding DFT-quality formation energies as provided by Faber et al. A random forest was the first method that was used for the predictions. As the performance of the random forest was rather poor when it came to classifying the stable elpasolites, we moved on to a neural network. We constructed a neural network that used the Mat2vec embeddings of the A, B, C and D elements of the elpasolites as input. Very good results were obtained. However, when initializing the model parameters with random numbers, we obtained similar results. We thus conclude that the use of Mat2vec embeddings to predict the formation energy of elpasolites did not turn out to be helpful. Nonetheless, this does not undermine the results of our neural network, which was able to achieve an almost perfect score on classifying stable and unstable elpasolites after prediction of the formation energy.

As we have seen in this chapter, our neural network did not 'need' the embeddings in order to make good predictions. They are simply very efficient at learning and are able to capture complex relationships between data points by themselves. In the next chapter, we will exploit the context-based nature of the embeddings to see if they can be helpful when mining a database of materials. More specifically, we will try to find good candidates for thermoelectric purposes with the help of embeddings.

Chapter 4

Predicting thermoelectric materials with embeddings

In this final chapter, we will try to predict thermoelectric materials with embeddings. As opposed to the previous chapter, we will not use a neural network to do this. Instead, we will rely on the contextual nature of the embeddings in the Word2vec model. Closely related words will be in close proximity in the 200-dimensional embedding space, whilst completely different words will be further apart. The distance between word vectors can be expressed by means of the cosine distance or cosine similarity, i.e. the cosine of the angle between them. As discussed in Section 1.3.2, Tshitoyan et al. were able to predict new thermoelectric materials by calculating the cosine similarity between the word 'thermoelectric' and the embeddings of materials. Thermoelectrics are materials that efficiently convert heat into electric power, making them interesting candidates to recover waste heat for useful purposes.

We will implement the same approach to predict thermoelectric materials in an in-house database from the Center for Molecular Modeling (CMM) at Ghent University. The database contains about 10,000 quaternary materials, which are promising thermoelectric candidates as they fulfill one of the necessary conditions: they have a sufficiently complex crystal structure. The objective of this chapter is to rank the quaternary materials by order of their expected thermoelectric performance with the help of embeddings. However, as the embeddings for the materials in the CMM database are not included in the Mat2vec model, we will look for two alternative approaches to construct material embeddings.

In order to test the validity of these approaches, we will compare our results to those of Jasper De Witte. In his master's thesis at the CMM, he constructed a deep learning model with ab-initio data to predict the thermoelectric power factor of materials. As we know from Section 1.3.2, the thermoelectric power factor is able to capture the performance of thermoelectric materials. The higher it is, the more suitable a material is for thermoelectric purposes. Jasper also applied his deep learning model to the materials in the CMM database. With the predictions of the thermoelectric power factor, Jasper was able to identify the 99 best thermoelectric materials in the CMM database. In what follows, we will investigate how many materials in the top 99 we are able to identify by using two approaches to construct embeddings for the materials.

4.1 The CMM database

The CMM database contains 10,527 quaternary materials and was constructed over the years by the Center for Molecular Modeling (CMM). However, we will not retain all of the materials. As we are specifically looking for usable thermoelectric materials, we applied a selection procedure beforehand. The same selection criteria were applied by Jasper De Witte on the database during his master's thesis.

First of all, materials likely to be thermodynamically unstable were removed. In the previous chapter, the formation energy was used to distinguish stable from unstable materials. However, as was briefly mentioned, the same chemical composition can possibly have a more negative formation energy for other crystal structures. For this reason, the database does not mention the formation energy, but instead the energy above the convex hull (E_{hull}) calculated with DFT for each quaternary material. The convex hull connects the most stable configurations for a given fraction of chemical elements. It is the energy distance to the convex hull that determines the true stability of the material. A stable material has an E_{hull} of zero as it lies precisely on the convex hull. The larger E_{hull}, the less likely the material will be stable in nature[29].

An example for Al-Fe is given in Figure 4.1, taken from Jasper De Witte's master's thesis. The $AlFe_2$ compounds with varying crystal structures, for example, do not lie on the convex hull and are therefore unstable. Instead, they will decompose to a combination of $AlFe_3$ and AlFe as these are the closest stable materials on the convex hull.

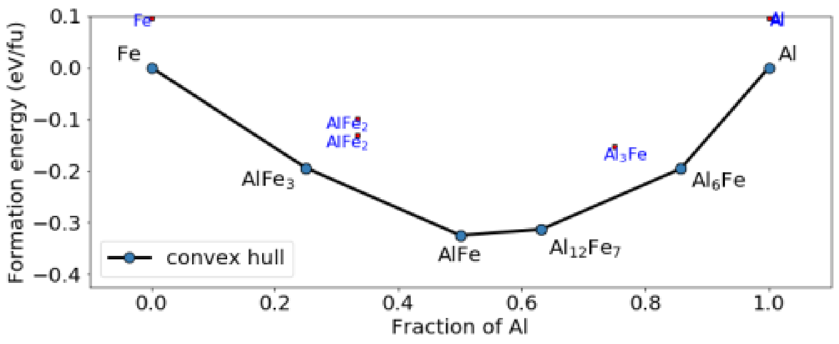


FIGURE 4.1: Example of a convex hull diagram for Al-Fe alloys. Blue dots represent stable compounds[29].

In the database of quaternary materials, we removed all of the materials for which $E_{hull} > 0.050$ eV/atom. This serves as a safety margin for numerical errors, systematic deviations of the simulation method and the possibility of metastable materials.

We also selected on the band gap E_g of the materials, also calculated with DFT for the materials in the database. The existence of a non-zero band gap is of great importance for thermoelectric materials. As the thermoelectric power factor is calculated as σS^2 with σ the electrical conductivity and S the Seebeck coefficient, a lot depends on the Seebeck coefficient. It was empirically found that the Seebeck coefficient for semiconductors, which have a non-zero band gap, is orders of magnitude larger than for metals (no band gap)[30]. For this reason, we removed the quaternary materials with a band gap under 0.050 eV as band gaps smaller than 0.050 eV become almost indistinguishable from metals.

After applying these selection criteria on the database, we end up with 1,198 materials and nine different stoichiometries with general formulae $ABCD_4$, $A_6BC_4D_3$, A_2BCD_4, $A_{18}B_6CD_{14}$, $A_8B_{24}CD_3$, $A_7B_6CD_{16}$, $A_4B_2CD_4$, A_2BCD_2 and $ABCD_3$. A histogram with the occurrence of each stoichiometry in the database is included in Appendix B, along with an overview of the A, B, C and D elements.

4.2 Approach

Our aim is to calculate the cosine similarity of the embedding vectors with the word 'thermoelectric' for the remaining 1,198 materials in the CMM database. The manner by which the embeddings were constructed will be discussed shortly, but first we will explain the idea behind calculating the cosine similarity to predict applications of materials.

The cosine similarity of two vectors **A** and **B** is:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}. \tag{4.1}$$

Using only the positive space, the cosine similarity of two vectors is limited to the range $[0, 1]$. The closer to 1, the more alike two vectors are. As the Mat2vec model was trained specifically to maximize the similarity of the word vectors for words that are used frequently in the same context (i.e. they have the same meaning), we expect the cosine similarity to be high for similar words. The idea is now that various applications of materials are also encoded in the embeddings. For instance, a material that is popular for the fabrication of fuel cells will frequently be mentioned in this context. The Mat2vec model will pick up the co-occurrence of our example material with fuel cells, which will result in two word vectors that live in close(r) proximity in the word embedding space as compared to other materials. This feature can be exploited to link materials to applications. However, what Tshitoyan et al.[1] have found and what was discussed earlier in Section 1.3.2, is that we can predict new material applications for materials that were never explicitly mentioned together with a specific application.

We will exploit this approach to predict thermoelectric materials in the CMM database, but with one major difference. Tshitoyan et al. used the embeddings of the materials that were present in the Mat2vec model to predict new thermoelectric materials. Unfortunately, we can not follow the same approach as only 14 out of the 1,198 materials are included in the vocabulary of Mat2vec. These 14 materials are only mentioned 5 times or less in the entire text corpus that the model was trained on. As a consequence, their embeddings will be of poor quality. We thus have to find a way to retrieve qualitative embeddings for materials that are not included in the Mat2vec vocabulary.

An alternative to construct embedding vectors for the materials in the quaternary database is to add up the constituent element vectors. For example, the embedding vector for $K_2MgSnSe_4$ can be constructed as

$$\text{vector('K')} + \text{vector('Mg')} + \text{vector('Sn')} + \text{vector('Se')}. \tag{4.2}$$

This is one possible way to construct an embedding vector out of the constituent elements. However, the database contains various stoichiometries (see Figure B.1 in Appendix B). The above method assigns the exact same embedding vector to crystals that contain the same elements, but have a completely different stoichiometry. It would be interesting to compare the first method where we solely added up the constituent vectors to a method where the stoichiometry of the materials is explicitly taken into account. In order to construct an embedding vector that contains the stoichiometry, we can construct the embedding vector of $K_2MgSnSe_4$ as

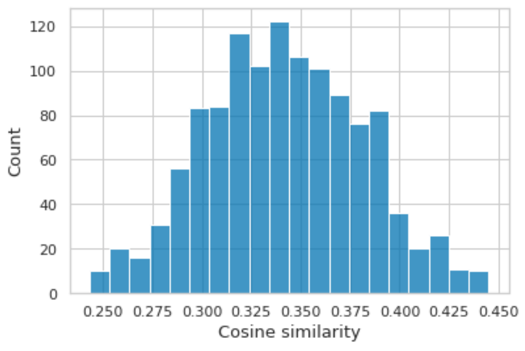
$$2 \cdot \text{vector}('K') + \text{vector}('Mg') + \text{vector}('Sn') + 4 \cdot \text{vector}('Se'). \tag{4.3}$$

Multiplying a vector with a positive scalar does not change its direction, but only its magnitude. By multiplying the word vectors of the elements with the respective stoichiometry number, we are able to give certain elements a higher weight in the final embedding vector.

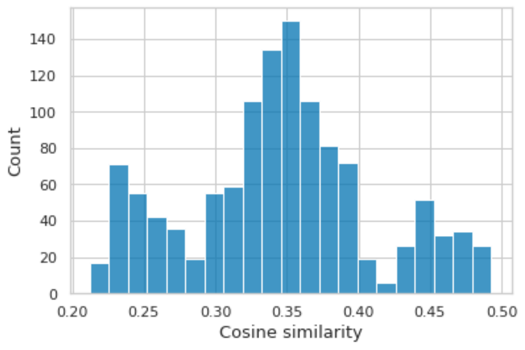
Both approaches will be used to construct material embeddings for the materials in the CMM database. Subsequently, a ranking of the materials can be made based on the cosine similarity of the word vector for 'thermoelectric' and the material embeddings. Before comparing our ranking to Jasper De Witte's results, the differences of both approaches to construct the embedding vectors are discussed.

4.2.1 Comparing different approaches to construct embeddings

For both approaches, we calculated the cosine similarity of the embeddings and the word vector for 'thermoelectric'. The results of these calculations are depicted on Figure 4.2 by means of a histogram with the distribution of the cosine similarity. On Figure 4.2a the embeddings were obtained by summation of the constituent elements, on Figure 4.2b the stoichiometry was explicitly taken into account. Embeddings without explicit multiplication for the stoichiometry, have a cosine



(A)



(B)

FIGURE 4.2: The results for the cosine similarity of the word 'thermoelectric' and the material embeddings. On 4.2a, the stoichiometry was not explicitly included in the material embeddings. On 4.2b, the stoichiometry was explicitly included in the embeddings.

similarity in between the range $[0.24, 0.44]$. With the stoichiometry of the crystals explicitly taken into account, we find a range of $[0.21, 0.49]$. The range of the cosine similarity thus seems rather similar for both approaches. Both histograms also exhibit a maximum count at a cosine similarity of about 0.35. Note that Figure 4.2a contains a lot of duplicates as materials with the same constituent elements but a different stoichiometry were assigned the same embedding vector.

Of course, the most important question is: are the highest ranked materials similar with both approaches? To answer this question, we ranked the materials in the database by decreasing order of cosine similarity with the word 'thermoelectric'. The 50 highest ranked materials for both approaches are included in Appendix B. Of these 50 materials, we find that 44 materials appear in both rankings. It thus seems that the different methods to construct the embeddings do not radically change the ranking of the materials. This is a positive result as we find that solely adding up the constituent elements of the quaternary materials does implicitly seem to capture stoichiometric information.

We will now investigate whether the embeddings were able to retrieve the same materials as Jasper De Witte found with his thermoelectric power factor predictions.

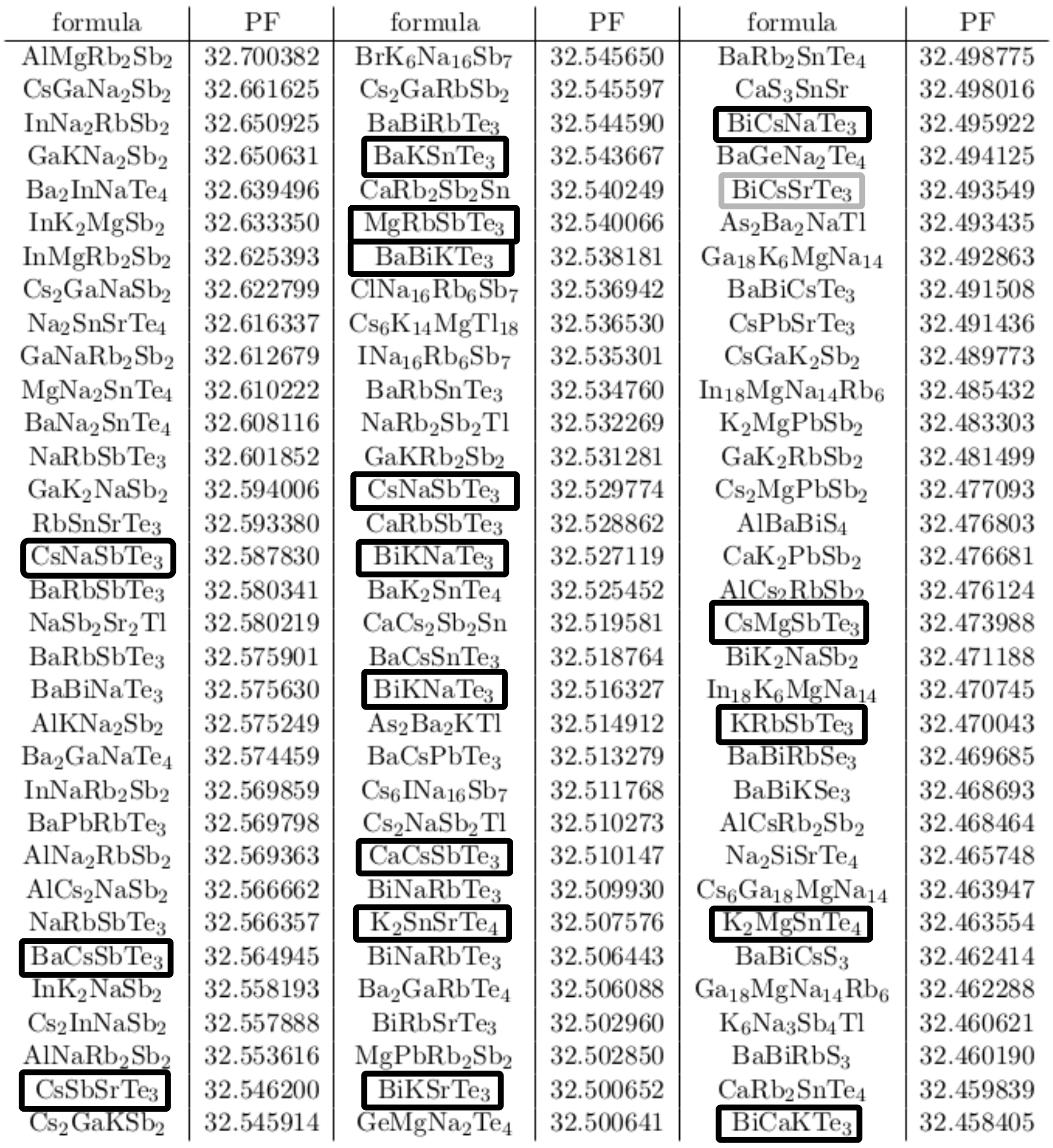
4.3 Comparison to Jasper De Witte's results

Jasper De Witte's top 99 of thermoelectric candidate materials is given in Figure 4.3. We also made a ranking of the 99 materials with the highest cosine similarities of the embeddings and the word 'thermoelectric', hereby using both methods with and without explicitly taking into account the stoichiometry in the embeddings. The materials in Jasper's top 99 that also appeared in our top 99 were indicated on Figure 4.3 with black borders. The method that didn't take the stoichiometry into account yielded a total of 17 matches in the top 99, the method that did take the stoichiometry into account matched with Jasper's ranking for the same 17 materials and one extra ($BiCsSrTe_3$, indicated with a grey border). At first, it looks disappointing that we did not find any of the fifteen highest ranked materials. Note however the small range in the predictions of the thermoelectric power factor. The difference in predicted thermoelectric performance between the first and the ninety-ninth material is not large.

How should we interpret this result then? If we were to randomly pick 99 materials out of the 1,198 that we started with, how many of them would be in the top 99? To get an idea of this number without having to do statistical combinatorics, we randomly picked 99 materials out of the database and calculated how many of them were in Jasper's top 99. After doing this a million times, we found on average 5.7 materials that matched with Jasper's top 99. Only 56 times in one million attempts did we find 17 or more matches. We thus find that the probability that our result with the embeddings was purely based on chance is 0.006 %.

4.4 Conclusion

In this chapter, we examined the potential of predicting thermoelectric materials in an in-house quaternary database of the CMM by calculating the cosine similarity with the word 'thermoelectric' and the embedding vectors of the materials. None of the materials in the database were (sufficiently) included in the Mat2vec vocabulary, so we decided to construct our own embeddings based on the constituent elements of the quaternary materials. Two approaches were tested. The first approach solely added up the elemental word vectors to obtain the word vector of the material. The second approach took into account the stoichiometry of the materials by multiplying



formula	PF	formula	PF	formula	PF
$AlMgRb_2Sb_2$	32.700382	$BrK_6Na_{16}Sb_7$	32.545650	$BaRb_2SnTe_4$	32.498775
$CsGaNa_2Sb_2$	32.661625	$Cs_2GaRbSb_2$	32.545597	CaS_3SnSr	32.498016
$InNa_2RbSb_2$	32.650925	$BaBiRbTe_3$	32.544590	$\boxed{BiCsNaTe_3}$	32.495922
$GaKNa_2Sb_2$	32.650631	$\boxed{BaKSnTe_3}$	32.543667	$BaGeNa_2Te_4$	32.494125
$Ba_2InNaTe_4$	32.639496	$CaRb_2Sb_2Sn$	32.540249	$\boxed{BiCsSrTe_3}$	32.493549
InK_2MgSb_2	32.633350	$\boxed{MgRbSbTe_3}$	32.540066	As_2Ba_2NaTl	32.493435
$InMgRb_2Sb_2$	32.625393	$\boxed{BaBiKTe_3}$	32.538181	$Ga_{18}K_6MgNa_{14}$	32.492863
$Cs_2GaNaSb_2$	32.622799	$ClNa_{16}Rb_6Sb_7$	32.536942	$BaBiCsTe_3$	32.491508
$Na_2SnSrTe_4$	32.616337	$Cs_6K_{14}MgTl_{18}$	32.536530	$CsPbSrTe_3$	32.491436
$GaNaRb_2Sb_2$	32.612679	$INa_{16}Rb_6Sb_7$	32.535301	$CsGaK_2Sb_2$	32.489773
$MgNa_2SnTe_4$	32.610222	$BaRbSnTe_3$	32.534760	$In_{18}MgNa_{14}Rb_6$	32.485432
$BaNa_2SnTe_4$	32.608116	$NaRb_2Sb_2Tl$	32.532269	$K_2MgPbSb_2$	32.483303
$NaRbSbTe_3$	32.601852	$GaKRb_2Sb_2$	32.531281	GaK_2RbSb_2	32.481499
GaK_2NaSb_2	32.594006	$\boxed{CsNaSbTe_3}$	32.529774	$Cs_2MgPbSb_2$	32.477093
$RbSnSrTe_3$	32.593380	$CaRbSbTe_3$	32.528862	$AlBaBiS_4$	32.476803
$\boxed{CsNaSbTe_3}$	32.587830	$\boxed{BiKNaTe_3}$	32.527119	CaK_2PbSb_2	32.476681
$BaRbSbTe_3$	32.580341	BaK_2SnTe_4	32.525452	$AlCs_2RbSb_2$	32.476124
$NaSb_2Sr_2Tl$	32.580219	$CaCs_2Sb_2Sn$	32.519581	$\boxed{CsMgSbTe_3}$	32.473988
$BaRbSbTe_3$	32.575901	$BaCsSnTe_3$	32.518764	BiK_2NaSb_2	32.471188
$BaBiNaTe_3$	32.575630	$\boxed{BiKNaTe_3}$	32.516327	$In_{18}K_6MgNa_{14}$	32.470745
$AlKNa_2Sb_2$	32.575249	As_2Ba_2KTl	32.514912	$\boxed{KRbSbTe_3}$	32.470043
$Ba_2GaNaTe_4$	32.574459	$BaCsPbTe_3$	32.513279	$BaBiRbSe_3$	32.469685
$InNaRb_2Sb_2$	32.569859	$Cs_6INa_{16}Sb_7$	32.511768	$BaBiKSe_3$	32.468693
$BaPbRbTe_3$	32.569798	Cs_2NaSb_2Tl	32.510273	$AlCsRb_2Sb_2$	32.468464
$AlNa_2RbSb_2$	32.569363	$\boxed{CaCsSbTe_3}$	32.510147	$Na_2SiSrTe_4$	32.465748
$AlCs_2NaSb_2$	32.566662	$BiNaRbTe_3$	32.509930	$Cs_6Ga_{18}MgNa_{14}$	32.463947
$NaRbSbTe_3$	32.566357	$\boxed{K_2SnSrTe_4}$	32.507576	$\boxed{K_2MgSnTe_4}$	32.463554
$\boxed{BaCsSbTe_3}$	32.564945	$BiNaRbTe_3$	32.506443	$BaBiCsS_3$	32.462414
InK_2NaSb_2	32.558193	$Ba_2GaRbTe_4$	32.506088	$Ga_{18}MgNa_{14}Rb_6$	32.462288
$Cs_2InNaSb_2$	32.557888	$BiRbSrTe_3$	32.502960	$K_6Na_3Sb_4Tl$	32.460621
$AlNaRb_2Sb_2$	32.553616	$MgPbRb_2Sb_2$	32.502850	$BaBiRbS_3$	32.460190
$\boxed{CsSbSrTe_3}$	32.546200	$\boxed{BiKSrTe_3}$	32.500652	$CaRb_2SnTe_4$	32.459839
Cs_2GaKSb_2	32.545914	$GeMgNa_2Te_4$	32.500641	$\boxed{BiCaKTe_3}$	32.458405

FIGURE 4.3: The 99 best performing quaternary materials in the CMM database, according to a prediction of the thermoelectric power factor performed by Jasper De Witte. PF denotes the power factor in $\ln(\mu W/cmK^2 s)$[29]. The materials that are indicated by black borders matched with our top 99 with both methods. The method that took into account the stoichiometry of the materials yielded one extra match, indicated by a grey border.

the elemental vectors with the corresponding stoichiometry number. With both approaches we found quite similar results.

Finally, we compared our result to Jasper De Witte's ranking of the materials that was based on the prediction of the thermoelectric power factor. In the top 99, we respectively found 17 matches and 18 matches for both methods to construct the embeddings, which is much more than a random sample would be able to achieve. Along with confirming that embeddings can be useful to predict material applications, this result further promotes the idea that quaternary materials may indeed harbor interesting candidates for application as thermoelectrics.

Chapter 5

Conclusion and outlook

5.1 Conclusions

Throughout this work, we have explored the potential of natural language processing (NLP) in materials science. More specifically, we applied the unsupervised NLP model Mat2vec that was constructed by Tshitoyan et al.[1] with 3.3 million abstracts of materials science articles. The resulting 200-dimensional word embeddings from the Mat2vec model were used in this work to investigate whether materials science knowledge can be extracted from them.

First of all, we started with the embeddings of chemical element names. A two-dimensional projection of the 200-dimensional embeddings was made to get some visual insight in the embedding space. On the two-dimensional projection, we were able to observe the clustering of elements from the same category in the periodic table. However, some elements seemed a little out of place, which was contributed to the contextual nature of the embeddings (such as the clustering of radioactive elements) and the fact that a lot of information was lost due to the two-dimensional projection of the 200-dimensional embeddings.

A more rigorous assessment of the embeddings of chemical elements was made by predicting several atomic and thermodynamic properties with a linear regression model. With 5-fold cross validation, we used the embeddings of the elements as input and the true value of a specific property as output for the linear regression model. As we only had 100 data points at most per property, given the limited number of chemical elements in the periodic table, the 200-dimensional embeddings were first reduced to 15 dimensions with principal component analysis (PCA). The R^2 values of the true values and the predictions for the atomic and thermodynamic properties varied between 0.45 and 0.75. When we compared this result to a similar procedure by Tshitoyan et al., we noticed that their results were noticeably better. However, the omission of certain elements before performing the linear regression procedure was the reason for this. As they were not clear in which embeddings they removed, we decided to use all of the elements and draw our own conclusions. One apparent problem that was identified by analyzing the results, was that very rare elements did not have qualitative embeddings. These poor embeddings can be ascribed to the fact that they almost never appeared in the materials science articles on which the Mat2vec model was trained. We can try to remove these from the dataset, but the problem is that we end up with even less data points. Furthermore, we noticed that the normalization of the word embeddings could be beneficial. This way, element names will refer to one unique embedding, instead of having multiple possibilities (e.g. ’tungsten’, ’wolfram’ and ’W’ all refer to the same element, but have different embeddings in Mat2vec). The normalization of the embeddings could be achieved by averaging over all possible embeddings that refer to one specific element name. Also in some cases, the opposite was true. For example, sometimes 'carbon' is mentioned in the context of diamond and sometimes it refers to graphite. One could try to separate these different contexts during the construction of the embeddings. Of course, the results in this chapter were obtained with a linear regression model. More complex non-linear models should be investigated to assess whether they are better suited to link the word embeddings of chemical elements to specific atomic and thermodynamic properties.

More complex models were applied in the third chapter of this work to predict the formation energy of elpasolites. Elpasolites are quaternary crystal structures with the general formula ABC_2D_6. We first tested the prediction of the formation energy by applying a random forest where the atomic numbers of the A, B, C and D elements served as input. As the formation energy distinguishes stable (negative formation energy) from unstable materials (positive formation energy), we were able to compare the classification of stable and unstable materials based on the predicted formation energy. The random forest struggled to correctly classify the stable elpasolites as most of the elpasolites in the dataset had a positive formation energy. Hence, the random forest is more likely to predict unstable elpasolites. Inspired by the positive results from Tshitoyan et al. to use embeddings as feature vectors in machine learning[1], we constructed a neural network with Mat2vec embeddings. We concatenated the embeddings of the A, B, C and D elements to serve as input in the neural network. Our neural network obtained an almost perfect score for the classification of stable and unstable materials. However, we wondered whether this was mostly due to the embeddings or due to the fact that neural networks are very efficient at learning. The latter seemed to be the case, as a neural network with randomly initialized embeddings was able to reach similar results. Some modifications were tried, such as averaging the embeddings of the A, B, C and D elements and unfreezing the embeddings (i.e. allowing the embeddings to be altered by the neural network), but no noticeable difference was observed with the neural network of the randomly initialized embeddings. We conclude from this that neural networks are simply able to capture complex relationships by themselves. Hence, the embeddings were not essential to make good predictions of the formation energy in our neural network. In order to test extensively the potential of embeddings in neural networks, one could look at different stoichiometries instead of solely taking into account the elpasolite structure. This hypothesis was tested for the quaternary materials of the in-house database that was used in the final chapter. While not included in this work, initial tests show no difference with the randomly initialized embeddings. However, the good news is that the inclusion of different stoichiometries did not lead to performance loss. This opens perspectives for larger databases where multiple stoichiometries are included. Also, another neural network architecture with a different number of layers could be implemented. It is most likely that the potential of the embeddings will become apparent when using less layers in the neural network. When there are less parameters to be optimized, the embeddings are allowed to play a more important part in the neural network. However, the question whether this will negatively impact the predictions is unclear and is left for future research.

In the final chapter, we examined the potential of embeddings to identify thermoelectric materials in an in-house database with quaternary materials. To do this, the cosine similarity between the word 'thermoelectric' and the embeddings of the materials in the database was calculated. Embeddings that yield a high cosine similarity are, as Tshitoyan et al. have shown[1], good candidates for thermoelectric purposes. As the materials in the database were not included in the vocabulary of Mat2vec, we applied two different methods to construct the material embeddings. One method did take the stoichiometry of the materials explicitly into account, the other did not. We found that both methods yielded similar results, which lead us to conclude that the stoichiometric information was somehow already contained in the embeddings. After comparing our results to those of thermoelectric power factor predictions previously performed by Jasper De Witte[29], we concluded that we were able to identify 17 materials that are expected to be good candidates for thermoelectric purposes. We thus obtain promising results. However, in order to truly validate this approach, we must test our method on databases that contain different material structures and also investigate other interesting applications (e.g. photovoltaics, superconductors,...).

5.2 Outlook

In this work, we have seen that NLP holds a lot of promise for the field of materials science. The performance of the elemental embeddings clearly showed they capture important chemical information. Using our simple machine learning models, high quality predictions on key properties for material discovery such as formation energies, were made. With the neural network that was constructed in this thesis, other material property predictions could also be performed. For example, zero versus non-zero band gap classification. The role of embeddings in these predictions can be further assessed and eventually be increased by altering the architecture of the neural network.

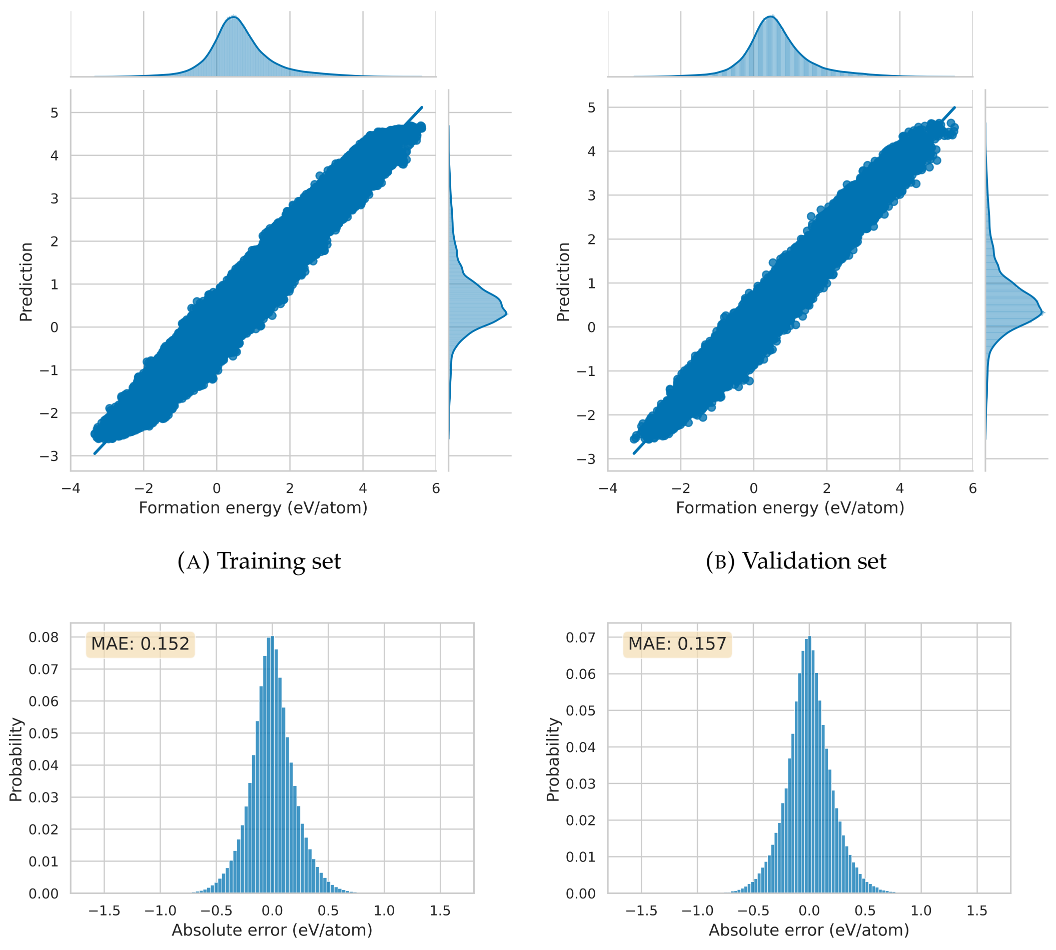
Various other applications of the Word2vec model, such as the classification of materials science articles and the generation of synthesis pathways for materials, were not explored in this work, but are applied extensively by other materials scientists. However, NLP continues to evolve outside of the materials science department. The breakthroughs and developments in NLP are occurring at an unprecedented pace. Recent developments in advanced context-aware embeddings stemming from deep learning, such as transformers, will likely be able to capture much more detailed information than the Word2vec model. As opposed to Word2vec, transformers process words in relation to all other words in a sentence. For example, Google's BERT framework (short for Bidirectional Encoder Representations from Transformers) is able to tell the difference between the different uses of the word 'running' in 'he is running a company' and 'he is running a marathon'.

One thing is for sure: transformers are already transforming the world of NLP, and will keep doing so in the future. This will surely bring along exciting applications for materials science, waiting to be discovered in future research.

Appendix A

Appendix for Chapter 3

A.1 Random forest



(A) Training set (B) Validation set

(C) Training set (D) Validation set

FIGURE A.1: Results on the training set and validation set for the predictions of the formation energy with the random forest.

A.2 Neural network with Mat2vec embeddings

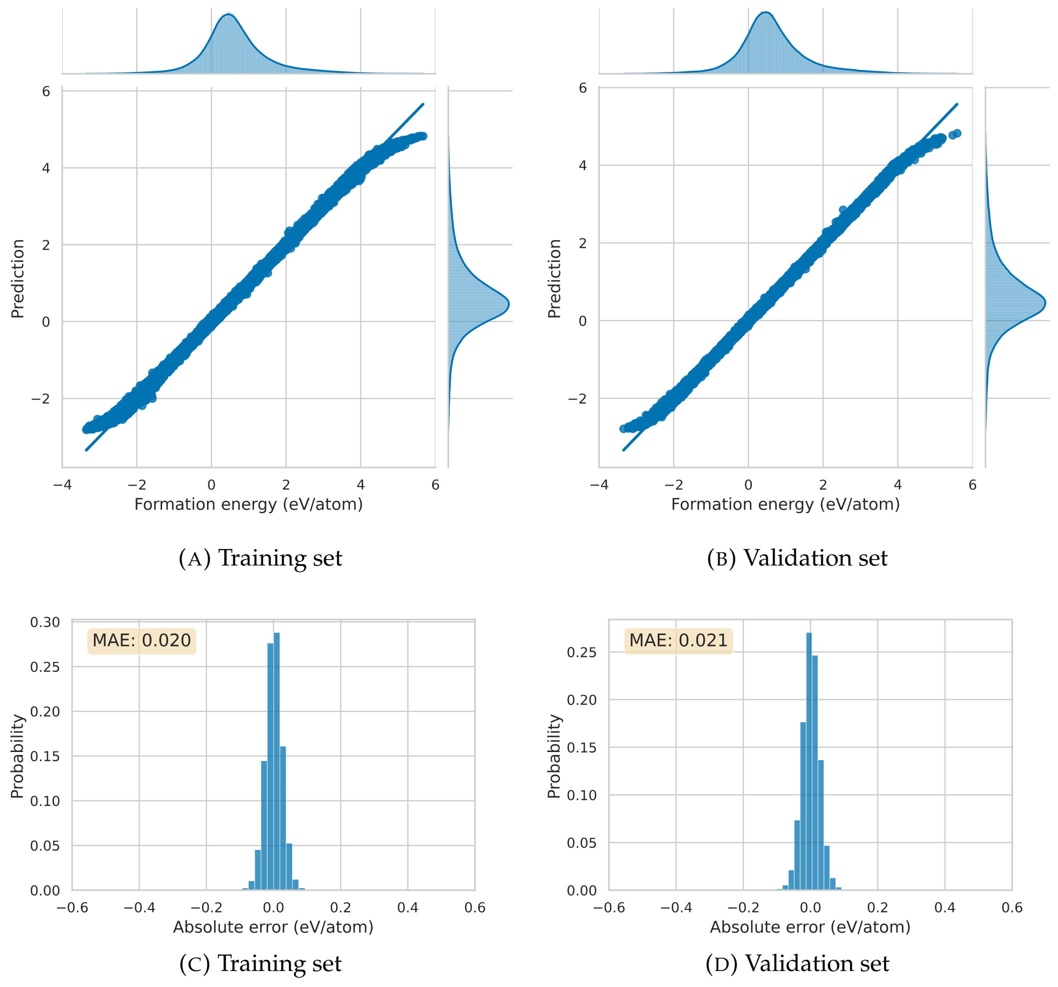
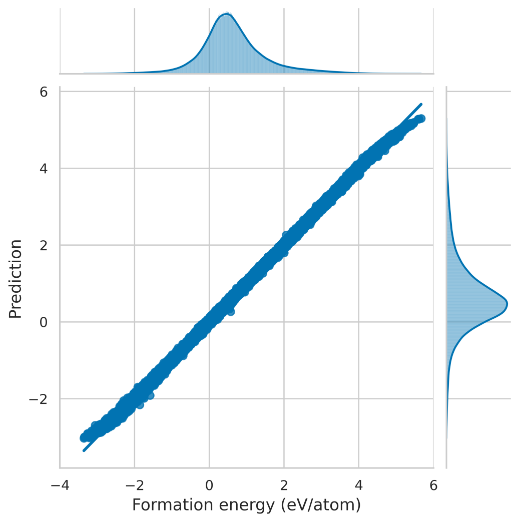
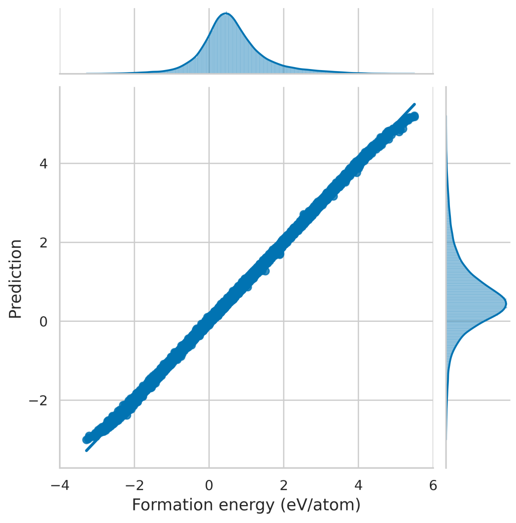


FIGURE A.2: Results on the training set and validation set for the predictions of the formation energy with the neural network with Mat2vec embeddings.

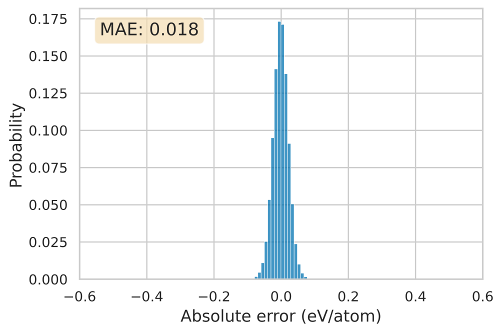
A.3 Neural network with random embeddings



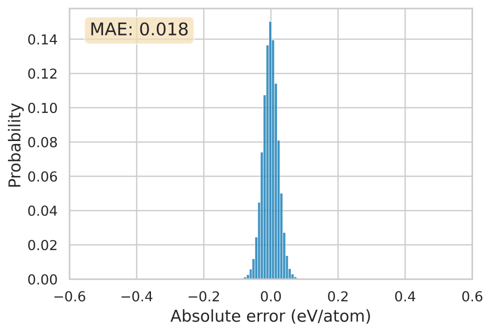
(A) Training set



(B) Validation set



(C) Training set



(D) Validation set

FIGURE A.3: Results on the training set and validation set for the predictions of the formation energy with the neural network with random embeddings.

Appendix B

Appendix for Chapter 4

B.1 The quaternary database after application of the selection criteria (E$_{hull}$ $<$ 0.050 eV/atom and E$_g$ $>$ 0.050 eV)

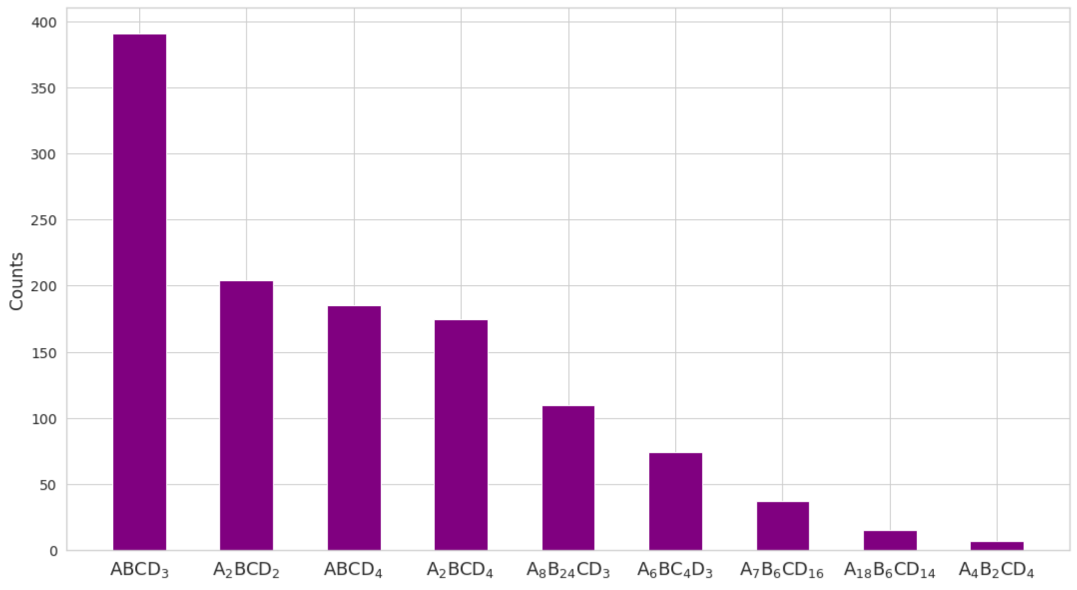


FIGURE B.1: The number of occurrences of each stoichiometry in the CMM database after applying the selection criteria.

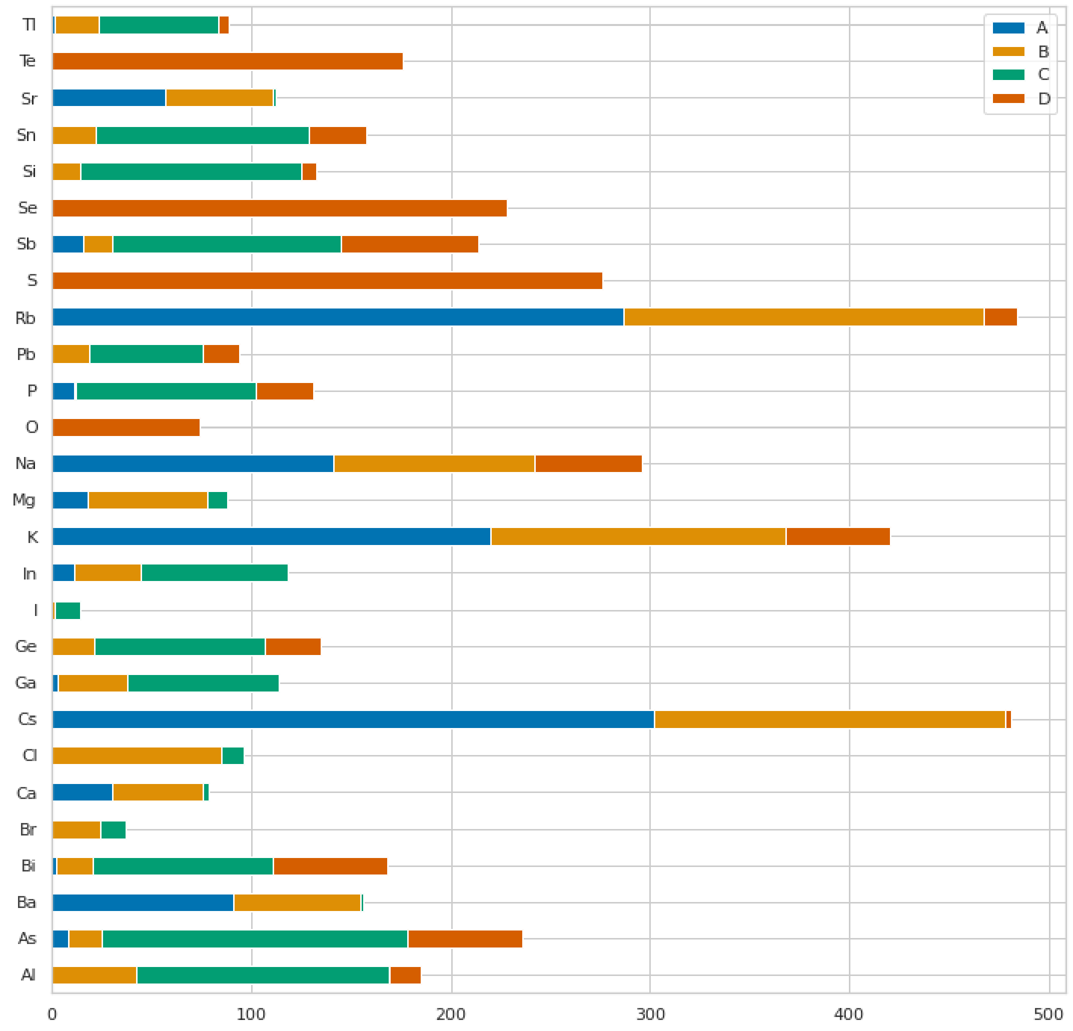


FIGURE B.2: The occurrence of the A, B, C and D elements in the CMM database after applying the selection criteria.

B.2 Top 50 of thermoelectric predictions with embeddings

Embeddings without stoichiometry				Embeddings with stoichiometry			
AsKNaTe3	0.445	K2SnSrTe4	0.424	AlKSiTe4	0.493	AlCsKTe3	0.480
CsKSbTe3	0.445	KSbSeSi4	0.424	AlKSiTe4	0.493	BiCaKTe3	0.479
BiKNaTe3	0.443	Ca2GeKTe4	0.423	AsKNaTe3	0.490	InRbSiTe4	0.479
BiKNaTe3	0.443	Ge2KMgTe4	0.423	CsInKTe3	0.488	Ge2KSrTe4	0.478
BiCsKTe3	0.440	BaBiKTe3	0.423	K2MgSnTe4	0.487	CsKSiTe3	0.478
BiCsKTe3	0.440	AlKNaTe3	0.422	Ca2KSnTe4	0.486	InKRbTe3	0.477
CsInKTe3	0.438	BiKSeSi4	0.422	CsInSiTe4	0.486	InKRbTe3	0.477
BiKMgTe3	0.438	Ba2KSnTe4	0.421	GaKSiTe4	0.486	AlCsSiTe4	0.477
CsKSnTe3	0.437	BaKSnTe3	0.421	BiKNaTe3	0.484	Ba2KSiTe4	0.477
CsKSnTe3	0.437	AsKRbTe3	0.421	BiKNaTe3	0.484	AsRbSiTe4	0.477
CsInSiTe4	0.433	BiGeKSe4	0.420	K2SnSrTe4	0.484	AsKRbTe3	0.476
BiCaKTe3	0.431	Ge2KSrTe4	0.419	BiKMgTe3	0.483	BiKSrTe3	0.476
GaKSiTe4	0.431	KRbSnTe3	0.418	K2MgSiTe4	0.483	Ba2GeKTe4	0.475
AlKSiTe4	0.431	KRbSnTe3	0.418	CsKSbTe3	0.482	AlCsGeTe4	0.474
AlKSiTe4	0.431	InKRbTe3	0.417	Ca2KSiTe4	0.482	AlCsGeTe4	0.474
KRbSbTe3	0.430	InKRbTe3	0.417	Ge2KMgTe4	0.482	KRbSbTe3	0.474
KRbSbTe3	0.430	CsGaKTe3	0.417	AlKNaTe3	0.482	KRbSbTe3	0.474
Ca2KSnTe4	0.429	AlCsGeTe4	0.416	BiCsKTe3	0.481	BiKRbTe3	0.473
CsKSiTe3	0.426	AlCsGeTe4	0.416	BiCsKTe3	0.481	BiKRbTe3	0.473
BiKSrTe3	0.426	GeKSSb4	0.416	CsKSnTe3	0.481	AlRbSnTe4	0.473
K2MgSnTe4	0.426	Ba2GeKTe4	0.416	CsKSnTe3	0.481	BaKSnTe3	0.472
BiKRbTe3	0.425	AsRbSiTe4	0.415	Ca2GeKTe4	0.480	BaBiKTe3	0.472
BiKRbTe3	0.425	InRbSiTe4	0.414	Ba2KSnTe4	0.480	KRbSnTe3	0.471
AlCsKTe3	0.424	CsMgSbTe3	0.414	K2SiSrTe4	0.480	KRbSnTe3	0.471
AlCsKTe3	0.424	Ca2KSiTe4	0.412	AlCsKTe3	0.480	CsGaSiTe4	0.470

TABLE B.1: Top 50 of the results of the cosine similarity with the word 'thermoelectric'. On the left, the stoichiometry of the embeddings was not taken into account. On the right, the stoichiometry was taken into account when constructing the embeddings.

Appendix C

Science communication

In order to communicate my master's thesis topic to the general public, I wrote an introductory article in Dutch about NLP for the website of the VVN (Vereniging voor Natuurkunde). The VVN is a Ghent University related student association that encourages science communication by organizing lectures and other activities about physics and astronomy. The article can be read on `https://vvn.ugent.be/blog/taalles-voor-computers/`.

Bibliography

[1] V. Tshitoyan, J. Dagdelen, L. Weston, A. Dunn, Z. Rong, O. Kononova, K. Persson, G. Ceder, and A. Jain, "Unsupervised word embeddings capture latent knowledge from materials science literature," *Nature*, vol. 571, pp. 95–98, 07 2019.

[2] P. Migdal, "king - man + woman is queen; but why?." `https://p.migdal.pl/2017/01/06/king-man-woman-queen-why.html`, 2017.

[3] L. Weston, V. Tshitoyan, J. Dagdelen, O. Kononova, A. Trewartha, K. A. Persson, G. Ceder, and A. Jain, "Named entity recognition and normalization applied to large-scale information extraction from the materials science literature," *Journal of Chemical Information and Modeling*, vol. 59, no. 9, pp. 3692–3702, 2019. PMID: 31361962.

[4] L. Bornmann and R. Mutz, "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references," 2014.

[5] A. Oberoi, "7 interesting applications of natural language processing (NLP)." `https://bit.ly/3bvXnSZ`, 2020.

[6] A. Medelyan, "8 natural language processing (NLP) examples you use every day." `https://bit.ly/3bv4RWr`, 2016.

[7] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, p. 1137–1155, Mar. 2003.

[8] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013.

[9] T. Mikolov, W. tau Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," 2013.

[10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013.

[11] D. Morgan and R. Jacobs, "Opportunities and challenges for machine learning in materials science," *Annual Review of Materials Research*, vol. 50, p. 71–103, Jul 2020.

[12] E. Kim, K. Huang, A. Tomala, S. Matthews, E. Strubell, A. Saunders, A. McCallum, and E. Olivetti, "Machine-learned and codified synthesis parameters of oxide materials," *Scientific data*, vol. 4, p. 170127, 2017.

[13] E. Kim, Z. Jensen, A. Van Grootel, K. Huang, M. Staib, S. Mysore, H. S. Chang, E. Strubell, A. McCallum, S. Jegelka, and E. Olivetti, "Inorganic Materials Synthesis Planning with Literature-Trained Neural Networks," *Journal of Chemical Information and Modeling*, 2020.

[14] C. Kang, H. Wang, J.-H. Bahk, H. Kim, and W. Kim, *Thermoelectric Materials and Devices*, pp. 107–141. 01 2015.

[15] Bokeh Development Team, "Bokeh: Python library for interactive visualization." https://bokeh.org/, 2020.

[16] L. McInnes, J. Healy, N. Saul, and L. Grossberger, "UMAP: Uniform Manifold Approximation and Projection," *Journal of Open Source Software*, vol. 3, p. 861, 09 2018.

[17] L. Mentel, "Mendeleev – a python resource for properties of chemical elements, ions and isotopes." https://github.com/lmmentel/mendeleev.

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] K. P. F.R.S., "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[20] Nanoshel, "Boron carbide." https://www.nanoshel.com/boron-carbide.

[21] S. Kirklin, J. Saal, B. Meredig, A. Thompson, J. Doak, M. Aykol, S. Rühl, and C. Wolverton, "The open quantum materials database (OQMD): Assessing the accuracy of dft formation energies," *npj Computational Materials*, vol. 1, p. 15010, 12 2015.

[22] F. A. Faber, A. Lindmaa, O. A. von Lilienfeld, and R. Armiento, "Machine learning energies of 2 million elpasolite ABC_2D_6 crystals," *Phys. Rev. Lett.*, vol. 117, p. 135502, Sep 2016.

[23] K. Biswas and M.-H. Du, "Energy transport and scintillation of cerium doped elpasolite Cs_2LiYCl_6: Hybrid density functional calculations," *Physical Review B*, vol. 86, 05 2012.

[24] S. Gražulis, A. Daškevič, A. Merkys, D. Chateigner, L. Lutterotti, M. Quirós, N. R. Serebryanaya, P. Moeck, R. T. Downs, and A. Le Bail, "Crystallography open database (COD): an open-access collection of crystal structures and platform for world-wide collaboration," *Nucleic Acids Research*, vol. 40, no. D1, pp. D420–D427, 2012.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[26] F. Bre, J. Gimenez, and V. Fachinotti, "Prediction of wind pressure coefficients on building surfaces using artificial neural networks," *Energy and Buildings*, vol. 158, 11 2017.

[27] J. Howard, "fastai." https://github.com/fastai/fastai, 2020.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32* (H. Wallach, H. Larochelle,

A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), pp. 8024–8035, Curran Associates, Inc., 2019.

[29] Jasper De Witte, "Data-efficient discovery of thermoelectric materials using deep learning," master's thesis, Universiteit Gent, 2020.

[30] K. Choudhary, K. F. Garrity, and F. Tavazza, "Data-driven discovery of 3d and 2d thermoelectric materials," *Journal of Physics: Condensed Matter*, vol. 32, no. 47, p. 475501, 2020.

[31] J. Howard and S. Gugger, *Deep Learning for Coders with fastai and PyTorch*. O'Reilly Media, 2020.

[32] E. Olivetti, J. Cole, E. Kim, O. Kononova, G. Ceder, T. Han, and A. Hiszpanski, "Data-driven materials research enabled by natural language processing and information extraction," *Applied Physics Reviews*, vol. 7, p. 041317, 12 2020.