Information extraction and machine learning for archaeological texts

Brandsen, A.; Gonzalez-Perez, C.; Martin-Rodilla, P.; Pereira-Fariña, M.

Chapter 11
Information Extraction and Machine Learning for Archaeological Texts

Alex Brandsen

Abstract Archaeologists are creating ever-increasing amounts of textual data. So much in fact, that manual reading and inspection has become practically impossible. By leveraging computational approaches, it is possible to extract relevant information from this big data, allowing for more efficient research and new analyses. In this chapter, methods and techniques to extract information from archaeological texts through Machine Learning are introduced and discussed, with a focus on practical examples. After reading the chapter, you should have a clear grasp on the possibilities of text mining in archaeology, the current state of research, and enough information to start your own text analyses.

Keywords Information extraction · Text mining · Machine learning · Data science

11.1 Introduction

In the last ten years or so, archaeologists have started generating 'big data': information assets characterised by the four V's: Volume, Velocity, Veracity, and Variety. Volume simply means the size of the data, generally meaning many gigabytes or terabytes of data. The Velocity is the speed at which data updates, and Veracity is a measure of how trustworthy data is, these V's are generally less relevant to archaeology. Variety speaks to the level of heterogeneity in the data, and how fuzzy or unclear data is, something we do encounter regularly in archaeology.

In short, big data is so unwieldy that it is not feasible to analyse it with conventional methods. This problem of having too much data has been described by multiple authors, with Bevan calling it a "data deluge" (Bevan, 2015, p. 1) and Vince noting "we are drowning in our own data" (Vince, 1996, p. 1). Dealing with

A. Brandsen (✉)
Faculty of Archaeology, Leiden University, Leiden, Netherlands
e-mail: a.brandsen@arch.leidenuniv.nl

C. Gonzalez-Perez et al. (eds.), *Discourse and Argumentation in Archaeology: Conceptual and Computational Approaches*, Quantitative Archaeology and Archaeological Modelling, https://doi.org/10.1007/978-3-031-37156-1_11

structured data—such as databases and geospatial data—has received a fair share of our attention, but much less research is being done on processing and analysing unstructured information: the documents that archaeologists write (Bevan, 2015).

These texts do contain a wealth of information, and by using computational tools to access, extract, and combine information in the documents, we can perform new synthesising research on large scales. Due to the amount of text data, computational methods almost become a necessity: in the Netherlands alone more than 4000 excavation reports are produced each year, not to mention thousands of books, papers, and preprints as well. When we extrapolate that to the situation across the world, it quickly becomes clear that manual inspection of these texts is unfeasible.

In this chapter, methods and techniques to extract information from archaeological texts through Machine Learning are introduced and discussed, with a focus on practical examples. After reading the chapter, you should have a clear grasp on the possibilities of text mining in archaeology, the current state of research, and enough information to start your own text analyses.

11.2 Information Extraction Techniques

In this section, an overview is given of techniques that are useful for Information Extraction. The focus here is on explaining what the methods do and what use they have, while the following sections go more into the technical details on how to practically apply these methods. The first part of this section explains some general concepts, and the subsections deal with specific techniques.

Natural Language Processing (NLP) is a research field which explores how computers can be used to understand and manipulate natural language, i.e. speech and written text in human language (as opposed to formal/constructed language such as programming languages) (Chowdhury, 2005). A document collection that we can analyse with NLP is called a **corpus**.

Text Mining is a subfield of NLP, and is a group of tasks all related to analysing written text (Feldman & Sanger, 2007). The most common task is **Information Extraction** (IE): extracting information from unstructured text.

In essence, IE is text simplification: turning unstructured text into a structured view of the information present in the text. There are a number of techniques that fall under IE, we here list the most used ones:

- **Named Entity Recognition** (NER), detection of entities (or concepts) in text. For example, finding all archaeological artefacts or time periods mentioned in a document.
- **Document Classification**, the process of automatically assigning labels to a text. For example, assigning subject metadata to a document by classifying it into one, or a number of categories.
- **Topic Modelling**, automatically clustering documents into distinct groups based on their content.

- **Relationship Extraction**, the identification of relations between entities. For example, finding relations such as [artefact] is found in [context].
- **Coreference Resolution**, detection of coreference between entities. For example, in the sentence "We found an arrow head, it is dated to the Neolithic", it is useful to know that "it" and "arrow head" refer to the same real world entity.
- **Terminology Extraction** or ontology extraction, a method of automatically constructing a thesaurus by analysing a large corpus.

At the moment in archaeology, NER, document classification, and topic modelling are being researched the most, and we will focus mainly on these techniques so we can illustrate the methods with archaeological examples.

Machine Learning (ML) is a form of Artificial Intelligence, which uses relations between data points in large data sets to create statistical models which can be used for various purposes. Generally, a Machine Learning algorithm will be able to take a human-annotated set of data (e.g. labelled entities in text) and create a statistical model which can predict new, unlabelled data (e.g. predict entities in text). Another way to make predictions is to use handcrafted rules: a **Rule-Based** approach. Here, an expert manually creates rules that can predict labels, e.g., "if a word is in an artefact word list, label it as an artefact". Both approaches have been used with various degrees of success in archaeological text mining. More detailed information about ML can be found in Sect. 11.5.

The term **Grey Literature** is used to describe documents which are not published in the traditional sense of the word by academic or commercial publishing houses, such as field reports and theses. A lot of research in archaeological text mining is focused on using this type of literature, as it is generally the most prevalent and the least studied.

11.2.1 Named Entity Recognition

Named Entity Recognition is the process of finding different categories of named entities (or concepts) in text. Quite often, the categories of entities are persons, organisations, locations, time periods, and quantities, as defined in CoNLL-2002 (Conference on Natural Language Learning), the most used NER benchmark (Tjong Kim Sang, 2002). For archaeology, these entities are not as relevant, except for time periods and locations. Generally, archaeologists are interested in entity types such as artefacts, materials, contexts, species, locations, and time periods. An example sentence with marked archaeological named entities is shown in Fig. 11.1.

NER can be useful for a range of applications. In archaeology, it is mainly used to automatically generate metadata, i.e. descriptions of data (Jeffrey et al., 2009; Byrne & Klein, 2010; Vlachidis, 2012; Niccolucci & Richards, 2013; Vlachidis et al., 2017). A lot of archaeological texts have limited or no metadata at all, and to be able to find these texts for research, it is useful to have some description of the data, such as which time periods, places, and artefacts are mentioned in the text. Instead of

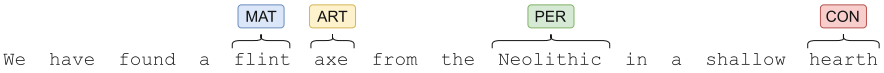

Fig. 11.1 Example sentence with named entities marked. Entity types have been shortened: MAT = material, ART = artefact, PER = time period, and CON = context

using all the detected entities, often a selection is made of the most important ones to serve as metadata. It is also possible to connect the detected entities to entries in thesauri, to further improve interoperability between data sets (Tudhope et al., 2011).

Instead of using a selection of entities as metadata, it is also possible to index all the entities in a search engine, together with the full text of the documents (Brandsen et al., 2019, 2020). This makes it possible for researchers to do advanced searches and find more relevant documents to their research. Recent research by Brandsen and Lippok (2021) shows that using such an intelligent search engine leads to more data and new insights. With either search on entities, or automatic metadata generation, the goal is to make the data more FAIR (Findable, Accessible, Interoperable and Reusable, Wilkinson et al., 2016).

Another approach that is made possible by entity extraction is pattern mining: using algorithms to automatically extract meaningful patterns from data. In other domains this has been researched extensively, but in archaeology it is quite rare. One example of pattern mining is the work by Wilcke et al. (2019), but they worked with hand-created XML (eXtensible Markup Language) data, not extracted entities. Their results are perhaps a bit lacking, but this might be partly due to the small amount of data. Once these methods are applied to thousands or millions of entities extracted from big data, more meaningful patterns might emerge.

11.2.2 Document Classification

Unlike NER where extraction of entities is the goal, document classification aims to assign one or more labels to a text. But similar to NER, this is often done to create metadata (Brandsen & Koole, 2021). Another approach is to label documents as relevant or irrelevant for a particular research question (Fischer et al., 2021). In archaeology, the focus has mainly been on NER, so there are not many examples. However, there are many possible applications, for example: determining whether or not tweets or reviews are about (a particular kind of) archaeology, or classifying a large amount of papers into certain categories for further study.

There are three variants of document classification:

1. Binary classification, each document is classified as either belonging, or not belonging, to one class (is a document relevant or not?)
2. Multi-class classification, each document can be classified as belonging to one of multiple classes (Which time period is a document about?)

3. Multi-label classification, each document can be classified as belonging to one or more classes (which subject(s) are discussed in this document?)

Generally, when assigning metadata, we would be looking for multiple classes for each document, so multi-label classification is the most common.

A particular type of document classification worth mentioning here is sentiment analysis, a task where the goal is to determine whether a text is positive or negative about a certain topic (Turney, 2002). This task is quite popular, receiving a lot of research interest, mainly in eCommerce and social media settings, where finding out whether a post or review is positive or negative is useful information. In an archaeological setting, it has been used to e.g. study the reactions to the destruction of heritage sites by ISIS (Cunliffe & Curini, 2018) and how tourists interact with monuments (Paolanti et al., 2019).

11.2.3 Topic Modelling

Topic modelling is a Machine Learning technique that can be used to cluster a collection of documents into groups, based on the word content of those documents. It is an unsupervised Machine Learning technique, as it does not require data annotated by humans (see Sect. 11.5.1). Because of this, it is a quick and easy way to start analysing a corpus. However, it is difficult to get accurate or meaningful results, which is why document classification is often more worthwhile. That being said, some potential uses for topic modelling include automatically grouping papers about a certain topic into subtopics to decide which will be manually read, and investigating changes in language use (or even theoretical trends) in archaeological literature over time and/or space (Plets et al., 2021; Jackson et al., 2020). An example of a topic model is shown in Fig. 11.4.

11.2.4 Information Retrieval

Related to Information Extraction is Information Retrieval (IR): methods to retrieve a set of documents based on a user defined query. In essence, IR is building search engines. IR is a research area of its own, with conferences and journals dedicated to the topic, and an in depth discussion is out of the scope of this chapter. However, it is worth briefly discussing IR in the context of Information Extraction.

Finding relevant literature for research is of course a common problem across all of science, and archaeology is no exception. But currently, most literature search is done using metadata search: searching through the title, description, and sometimes keywords manually entered by the author or archival service. Such metadata can not fully capture all the information present in a document, and as such relevant information can be missed. More advanced search systems, including full-text

search and named entity search, have been explored to some extent (Paijmans & Brandsen, 2010; Gibbs & Colley, 2012; Brandsen et al., 2019), but more research is needed to create better search engines for archaeologists.

11.3 Previous Research on Information Extraction in the Archaeology Domain

As mentioned by Richards et al. (2015), archaeological texts have excellent potential for text mining, due to its relatively well-controlled vocabulary. Much work has gone into producing thesauri (controlled word lists) in multiple languages (Gilman & Newman, 2007; Brandt et al., 1992), which we can leverage to extract information from text. In the last fifteen years, a range of projects have been undertaken which have attempted to use text mining within archaeology, starting with rule-based methods, and gradually moving towards Machine Learning based methods. In this section, we provide a brief overview of these text mining studies.

Amrani et al. (2008) created a workflow allowing archaeologists to extract information from English texts, but in a quite specialised way on a small collection. At the same time, The OpenBoek project (Paijmans & Brandsen, 2010) used Machine Learning to automatically label time periods and locations in Dutch field reports, which were searchable together with the full text in a web application. Byrne and Klein (2010) experimented with extracting archaeological events and converting them to RDF (Resource Description Framework) triples, to increase the interconnectivity between data sets from different sources.

The Archaeotools project used a combination of rule-based and Machine Learning approaches to automatically generate location, time period, and subject metadata for a small selection of reports. This generated metadata could then be used for searching in a facetted interface (Jeffrey et al., 2009). In the OPTIMA project, Vlachidis (2012) applied rule-based techniques to perform NER and express entities in the CIDOC-CRM schema.[1] The output of this research was further built upon in the STAR and STELLAR projects, where Tudhope et al. (2011) created a search demonstrator which searches through extracted entities from text and five excavation databases at the same time.

As part of the international ARIADNE project, some experiments were undertaken with NLP on grey literature. The ADS (Archaeology Data Service) in the UK created a prototype web application which uses NER to automatically create metadata for English reports, and experimented with rule-based NER for Dutch and Swedish reports as well (Vlachidis et al., 2017).

[1] The International Documentation Committee—Conceptual Reference Model (a way to model information) for cultural heritage and museum documentation, as defined by the International Committee for Documentation (CIDOC) (2014).

In her Master's thesis, Talboom (2017) specifically targeted zooarchaeological entities in reports, using Machine Learning to perform NER. Building on her work, Talks (2019) added more entity types and did an extensive evaluation with users.

Very recently, Fischer et al. (2021) used text mining as part of their research on ruralisation in the Netherlands. They created a term document matrix and compared this with a list of keywords related to the topic of ruralisation, to assess the usefulness of a large number of reports for a number of topics.

In a slightly different direction, Plets et al. (2021) describes research on grey literature from Belgium, looking at theoretical trends over time. They successfully manage to use text mining to find these trends and chart the decrease in text quality due to developer-led archaeology. Similarly, Jackson et al. (2020) used topic modelling techniques on English data to see if there are patterned ways in which archaeologists write about osteology.

In the Netherlands, the AGNES (Archaeological Grey literature Named Entity Search) project has been working to create a search engine for Dutch excavation reports, which leverages Machine Learning NER to make more efficient and detailed search possible (Brandsen et al., 2019). This project will be extended to also include English and German documents, and include more document types (such as books, papers, etc) over the next four years.

From this overview it is evident that there is a clear focus on grey literature, presumably due to their ubiquity and potential for Information Extraction. A lot of research also focuses on making data more FAIR (Wilkinson et al., 2016), by automatically creating more metadata, by building search engines, and by expressing unstructured text information into machine-readable formats to increase interoperability.

Generally, the aim is to assist archaeologists in their research by making big data sets that are difficult to navigate more manageable and searchable. The hope is that by harnessing computer power to analyse and summarise big data, we can do better synthesising research at large scales, leading to a better view of the past.

11.4 Preprocessing

As mentioned in the introduction, text is unstructured data. This means that there is no external structure added to the data which allows computers to easily process it. For example, in a database table, each number or string is stored in a cell, which is in a specific column and row. This column/row structure allows computers to 'understand' the data and perform analyses. Humans can easily make sense of text by reading it, because we have an incredible amount of background knowledge: we know the world we exist in, we know which words describe which concepts in that world, we know the language the text is written in, and we have the ability to read words and process them into meaningful information in our minds.

However, to computers, text is just a sequence of individual symbols with no inherent meaning, and they do not know the language or the concepts in the real

world that the text describes. This makes it a lot more difficult to work with text data than it is to work with structured data. To convert text into a format where a computer can work with it, we need to do some preprocessing. During this process, we can also help our analyses by excluding or transforming words (further detailed below). While preprocessing is not the most exciting part of a text analysis, the choices made in this part of the process can make big differences in the outcome of an analysis. In addition, it does tend to take up a substantial amount of time: often more time is spent on defining and fine-tuning the preprocessing methods than on the actual analysis itself. In the next couple of sections, an overview is given of common preprocessing tasks, how to perform them, and what effect they (can) have on the results of an analysis.

Note
Most of the software we reference in this chapter is Linux and Python based, but all steps and methods can be done with other software on other platforms as well.

11.4.1 Converting to Plain Text

For many data sets, the first step is to convert the files to plain text (.txt files). Most often, text data sets in archaeology are collections of PDF (Portable Document Format) or Microsoft Word files. They are not ideal for computation approaches as these formats also encode style information (among other details), which we generally do not need in our analyses and just cause unwanted noise.

A lot of tools exist to convert PDF files to plain text. Commonly used tools are `pdftotext`,[2] a command-line utility for Linux distributions, and the `PDFMiner`[3] and `PyPDF2`[4] packages in Python.

For Word files, the most used tool is `docx2txt`,[5] or the Python library `textract`.[6] which can extract text from a range of file formats, also including image and sound files. Choosing which tool is best for a use case depends on the end goal of the analysis, and which software you are using, but any tool that creates plain text should be sufficient.

[2] https://www.xpdfreader.com/pdftotext-man.html.
[3] https://pdfminersix.readthedocs.io/.
[4] https://pypi.org/project/PyPDF2/.
[5] http://docx2txt.sourceforge.net/.
[6] https://textract.readthedocs.io/en/stable/.

11.4.2 Optical Character Recognition

Most documents we deal with nowadays are 'born digital', which means they were created using computer software. Born digital documents will have the text encoded as actual characters which we can extract using the methods mentioned above. However, some files will be scanned pictures of existing hard copy documents, this is mainly the case for older documents (before the 2000s). In this case, the file does not contain actual computer readable characters, but just a grid of pixels in varying colours as far as the computer is concerned. To extract computer-readable text, the process of Optical Character Recognition (OCR) is needed (Merali & Smith, 1985). This method 'reads' the image of the text, and uses pattern matching and/or Machine Learning methods to translate these into machine-readable text. OCR is never 100% accurate, and as such you should expect noise being introduced in this phase, with the level of noise largely dependent on the quality of the original print and the quality of the scans. But once the computer readable text is available, we can continue with the rest of the preprocessing.

11.4.3 Sentence Boundary Detection

Most methods and analyses require one sentence per line in the text file, but often this is not what the plain text conversion provides. The first step is to do sentence boundary detection (also called sentence boundary disambiguation): automatically detecting where sentences begin and end (Riley, 1989). This might seem trivial, as sentences are normally ended by a full stop, exclamation mark or question mark, but in practice this is quite challenging due to the potential ambiguity of punctuation marks. A full stop for example, can be a part of an abbreviation, an email address, or be a decimal point, all instances where we should not end a sentence. The following sentences illustrate the problem:

> We found a Neolithic(?) flint axe in pit no. 2, but didn't find any pottery.
> An adjacent post hole yielded enough charcoal for a C14 dating.

Here, a number of potential problems are highlighted: the question mark after "Neolithic" and full stop after "pit no" are not the ends of the sentence. Also note the full stop on the next line, this is not a typo, but a common occurrence in text created by PDF conversion and/or OCR. The correct sentence split is on the full stop after "pottery":

We found a Neolithic(?) flint axe in pit no. 2, but didn't find any pottery.

An adjacent post hole yielded enough charcoal for a C14 dating.

Sentence boundary detection is mainly done by using rules of varying complexity, but can also be tackled by Machine Learning. In Python, the most commonly used method is the NLTK (Natural Language ToolKit) package, which also performs a large number of other NLP tasks (Bird et al., 2009).

Note
The first sentence in the box above will be used to illustrate all the following steps, to give a view of the full process.

11.4.4 Tokenisation

Like we mentioned earlier in this section, computers see text as a sequence of symbols with no inherent meaning. This also means that computers do not know what words are, or how to distinguish where a word starts and ends. To convert a sentence into a sequence of words, we use tokenisation, which returns a list of tokens. Tokens are similar to words, and a token often is a word, but not always. A token is defined as an instance of a sequence of characters that are grouped together as a useful unit for processing (Manning et al., 2008). This difference between words and tokens can be illustrated by tokenising our example sentence:

We found a Neolithic (?) flint axe in pit no . 2 , but did n't find any pottery .

In this example, most of the tokens are indeed words, but punctuation marks have also become individual tokens and "didn't" has been converted to two separate tokens. This tokenisation process is important as it removes noise from words (such as the brackets and question mark after 'Neolithic') and turns sentences into chunks of information that can be processed further.

11.4.5 Normalisation

Once we have a list of tokens, we can normalise and clean the text. There are a lot of different methods that can be applied at this stage, but the following most common steps are discussed: lowercasing, removing words, stripping characters, stemming, and lemmatisation.

Important
All normalisation preprocessing steps described below can affect the end result of the analysis both positively and negatively. Depending on the data, the methods used and the end goal, each normalisation technique should be individually considered.

11.4.5.1 Lowercasing

This is pretty much what the title suggest: changing all uppercase characters to their respective lowercase versions. Lowercasing is useful for most analyses, as it decreases the number of different tokens in your data set, and merges the uppercase and lowercase versions of a token into one. This intuitively makes sense as there is no semantic difference between e.g. "Axe" and "axe", but to a computer, these are two different strings, and will be analysed separately.

There are some exceptions in which case it is better to keep the uppercase characters, a good example is Named Entity Recognition (NER), a method for automatically finding and labelling certain concepts such as person names and place names. To be able to recognise such a name, having the casing intact is useful, as names will most often be capitalised, making it easier to distinguish between the last name "Flint" and the material "flint". Lowercasing is a common function in most text analysis software and programming languages, e.g. in Python the `lower()` function can be used. Here is our example sentence with lowercasing applied:

```
we found a neolithic ( ? ) flint axe in pit no . 2 , but did n't find any pottery .
```

11.4.5.2 Removing Words

Quite often, certain words are uninformative for an analysis, and removing them will reduce noise. It removes low-level information from the text to give more weight to important information. Besides this effect, removing common words also

reduces the size of the data, and thus reduces the training time of Machine Learning algorithms.

A method that is often used is to remove so-called 'stop words'. Stop words are the most common words in a language, like articles, prepositions, pronouns, conjunctions, etc. Some examples in English include "the", "a", "so", "is" and "that". The words that should be deleted are defined in a manually defined stop words list. Luckily, most—if not all—text analysis software provides such a list for English, and often many more languages too. Here, we have used the NLTK stop word list to remove them from our example:

```
found neolithic ( ? ) flint axe pit . 2 , find pottery .
```

Another way to reduce the total number of different tokens is to remove the n most common tokens, this is very similar to removing a predefined list of stop words. The other way around, it is also possible to remove tokens that only occur n times in the data set, with n often being a number between 1 and 3. This way we remove tokens that are uncommon, and thus uninformative for some tasks.

Do keep in mind that for certain types of analyses, having stop words or uncommon words in your data can be useful. An example is sentiment analysis, where words like "not" are indicative of a negative sentiment.

11.4.5.3 Stripping Characters

Besides removing words, we can also remove other types of tokens, such as punctuation, numbers and symbols. Doing all three on our example sentence leads to:

```
found neolithic flint axe pit find pottery
```

Quite often these types of tokens are not informative, but there are exceptions. If you are trying to find C14 dates in text, removing all symbols will also remove "±", which is a very strong indicator of a C14 date in archaeological texts.

11.4.5.4 Stemming

Stemming is the process of reducing words to their stem, i.e. removing the suffix of the word. For example, "house", "houses" and "housing" all have the same stem: "hous". As you can see, the stem does not need to be an actual word, although it often is. It is sufficient if all related words are reduced to the same stem—even if

that stem is not a word—as to a computer there is no semantic difference. Stemming groups related words into one representation, again reducing the variety in the data. If we apply stemming using the Porter stemmer (Porter, 1980) from NLTK to our example sentence, we end up with:

```
found neolith flint axe pit find potteri
```

While stemming in general does reduce the variety of tokens, in this case it has not: "found" and "find" have been assigned separate stems, while really they have a very similar meaning. Stemming does not take into account the actual meaning of a word, but uses rules to remove suffixes.

11.4.5.5 Lemmatisation

Lemmatisation is similar to stemming, but a bit more advanced. It reduces a word not to its stem, but to its lemma: the dictionary form of a word. Instead of chopping off a word's suffix, it uses linguistic features to determine the Part Of Speech (POS) and semantic meaning of a word, and subsequently finds the corresponding lemma. This means that the lemma of "axing" is "ax" and the lemma of "axe" is "axe", indicating the semantic difference between the two. If we use lemmatisation instead of stemming, our example sentence looks like this:

```
find neolithic flint axe pit find pottery
```

Here we see that "found" and "find" are both assigned the same lemma: "find", unlike with stemming. Depending on your application, either stemming or lemmatisation can be more appropriate, but something to keep in mind is that lemmatisation is a more difficult task than stemming, and as such is less accurate.

11.4.5.6 Normalisation and Information Loss

As already indicated with examples for e.g. lowercasing, stripping characters, and removing words, not all normalisation techniques are useful for every analysis. This is because the goal of normalisation is to reduce the complexity i.e., simplify data. However, when data is simplified, this means some information is lost. The trick is finding a balance between normalising the text to such an extent that classifiers can more easily learn statistical patterns, while not removing any information that might be useful for that classifier.

Generally, there are certain types of preprocessing that are commonly used for each type of analysis, but every data set is different and requires a thorough consideration by inspecting the data and comparing normalisation steps. In archaeology, we often deal with particular types of fuzziness and ambiguity in our data, when compared to other domains. This means that when working with archaeological data, careful consideration is needed from both the computer science side and the archaeology side, to make optimal choices regarding normalisation and other choices during the development of text mining tools.

11.4.6 Adding Structure

At this point we have preprocessed our text, and we are at the final step before we can start our analysis: adding structure, so a computer can do something with our data. The easiest way to do this is the so-called Bag of Words (BoW) approach (Manning et al., 2009). Here we simply create a table with a column for each word, and each row representing a sentence (or document). In the cells, we store how often a word occurs in each sentence. This word count is called Term Frequency (TF). See Table 11.1 for an example using the two sentences we introduced in Sect. 11.4.3. Note that the order of the words in the original sentences is lost, this is why it is called a Bag of Words: all the words end up in a 'bag', shuffled and without ordering. Most text analysis software will do this data transformation automatically.

While here the BoW is represented as a table for clarity, in reality each sentence is stored as a vector: a list of numbers. In Python, this would look like:

```
[0, 1, 0, 0, 0, 2, 1, 0, 1, 1, 0, 1, 0]
```

At this point, the computer does not know which TF stands for which word, because it does not need this information. Based purely on the vectors of a large number of sentences (or documents), it can extract statistical relationships and make predictions based on those. For example, if we are interested in automatically finding sentences about the Neolithic, an algorithm would infer that if the TF of 'neolithic' is not 0, it has the label Neolithic. Of course this is not a great example as just looking for the term 'neolithic' would be enough to find that out, but relationships between other (less literal) words can also be used to make predictions.

11.4.6.1 Term Frequency and Inverse Document Frequency

In the above example, we used the Term Frequency to create the vector. While this is an easy way to create a vector, it is not always ideal. Some words are simply more frequent in general, but that does not mean they are actually more important or relevant. To counteract this problem, we can use the Term Frequency–Inverse Document Frequency (TF-IDF), which lowers the value if a word occurs in many

Table 11.1 Example of two sentences converted into a Bag of Words table, after preprocessing

	Adjacent	Axe	c14	Charcoal	Dating	Find	Flint	Hole	Neolithic	Pit	Post	Pottery	Yield
1	0	1	0	0	0	2	1	0	1	1	0	1	0
2	1	0	1	1	1	0	0	1	0	0	1	0	1

documents (Manning et al., 2009). TF-IDF is currently the most used statistical measure for information retrieval and text mining (Beel et al., 2016).

11.4.7 Selecting Preprocessing Steps

All the preprocessing steps discussed here have different effects on the eventual input data for Machine Learning, and can greatly affect the outcome. It is always worth considering which steps will help for a particular analysis, as not all steps are always applicable.

In general though, when doing document classification and topic modelling, most of these steps will help increase the performance, as they decrease the variety in the text and group different forms of semantically similar words together, making it easier to generalise over the data. On the other hand, for NER (and also e.g. word embeddings, see Sect. 11.5.5), it is wise to only perform sentence detection and tokenisation and none of the normalisation steps, as differences in e.g. casing and symbols can be key indicators for entities. Lastly, another option is to simply try all possible combinations of preprocessing steps in a brute force method, and select the best performing combinations (Brandsen & Koole, 2021).

11.5 Machine Learning

Once the data has been selected, preprocessed, and converted into the right format, the actual analysis can be performed, i.e. NER or document classification. Most information extraction methods used today are based on Machine Learning (ML), a subfield of Artificial Intelligence. ML can be defined as the study of algorithms that automatically improve through experience (Mitchell, 1997). This means that these algorithms can build models based on training (or sample) data, without being programmed by a human to do so, in this way 'learning' by themselves how to predict labels for unseen data. Machine Learning is ubiquitous in modern life, being used in everything from predicting the spam status of emails to preventing traffic accidents in cars by automatically detecting obstacles.

Within archaeological research, ML is also becoming more popular, and is being used for a wide range of problems. Some examples are the automatic detection of archaeological features in LiDAR (Light Detection And Ranging) data (Verschoof-van der Vaart et al., 2020; Trier et al., 2018), classification of pottery types based on photos (Gualandi et al., 2021; Pawlowicz & Downum, 2021), analysing projectile point typology (Nash & Prewitt, 2016), and differentiating between lithic assemblages (Grove & Blinkhorn, 2020). For a more in depth overview of ML in archaeology and cultural heritage, see Bickler (2021) and Fiorucci et al. (2020).

Machine Learning has also been applied to textual data, both modern and ancient. Some examples of the analysis of ancient texts are the translation of

cuneiform script using an app (Sanders, 2018) and the reconstruction of missing pieces of ancient Greek text (Sommerschield, 2020). But mostly, ML is used to analyse modern texts about archaeology: e.g. books, papers, theses, and field reports written by archaeologists in the last couple of decades. Some examples include codifying semantically consistent definitions of archaeological concepts (Davis, 2020), Named Entity Recognition (Paijmans & Brandsen, 2010; Vlachidis et al., 2017; Tudhope et al., 2011; Talboom, 2017; Brandsen et al., 2019; Vlachidis et al., 2021), classifying reports on time period, location and/or subject (Jeffrey et al., 2009; Brandsen & Koole, 2021), topic modelling (Jackson et al., 2020), creating a list of relevant documents for certain topics (Fischer et al., 2021) and investigating theoretical trends over time (Plets et al., 2021).

Machine Learning is often juxtaposed with rule-based approaches: methods where a researcher defines a set of rules by hand, which are used to predict labels. These rule-based methods have been successful in many cases, but we see that ML approaches are being used more and more, as they are generally more effective at learning patterns in complex data (Richards et al., 2015; Bickler, 2021). This is also why this chapter will mainly focus on ML methods. That being said, rule-based approaches still have a place in current research, especially for problems where there is not a lot of training data, and can be used together with ML methods in many cases.

11.5.1 Supervised and Unsupervised Learning

Machine Learning can be subdivided into two main types: supervised and unsupervised learning. The difference is that supervised learning uses data that has been labelled by humans, while unsupervised learning uses raw, unlabelled data. In effect, supervised learning is where an algorithm learns patterns between the raw data and true labels, while unsupervised learning detects patterns in the raw data itself.

To give an example of **supervised learning** in text, we can take the automatic labelling of papers with topics. Imagine a stack of thousands of archaeology papers with no information on topic (no assigned keywords in the metadata). But it would be useful to know the topic, so we can make a selection of which papers to read. It is possible to create a classifier model to predict the topic (or class) of a paper, by feeding a supervised Machine Learning algorithm a collection of data that has been labelled by an archaeologist. See Table 11.2 for a simplified example with two possible subjects: Neolithic or Bronze Age. The first four rows are training data, with a label assigned by a human. The ‘Content’ column contains the titles of the papers, preprocessed as discussed in Sect. 11.4.

Table 11.2 Simplified example of Machine Learning document classification, with four human labelled training examples and one unlabelled document in the bottom row

Type	Content	Class (or label)
Training	Flint domestication use wear analysis	Neolithic
Training	Knapping flint wheat harvest	Neolithic
Training	Flint bronze sickle knapping	Bronze Age
Training	Bronze axe wood use wear analysis	Bronze Age
Prediction	Domestication flint knapping microscope	???

Table 11.3 Example prediction based on how often terms occur in a class in the training data. The percentages show in which proportion of documents from a class this term occurs. The label 'Neolithic' can be assigned with a 66.6% certainty

Term	Neolithic %	Bronze age %
Domestication	50%	0%
Flint	100%	50%
Knapping	50%	50%
Microscope	n/a	n/a
Average	66.6%	33.3%

Try it Yourself
Based on the information in this table, a human would be able to predict the label of the last row. If you want to do some human brain powered 'machine' learning, you can try it yourself: which label do you predict for the last row?

By reading the words in the examples, humans can figure out that the terms "domestication", "flint", and "knapping" are indicators that the predicted label should be "Neolithic", even if they have no prior knowledge of archaeology. Computers can do this too, but mathematically. Imagine each term being assigned a score between 0 and 1, based on which documents the terms occur in. A score of 0 means it only occurs in Neolithic, a score of 1 means it only occurs in Bronze Age, and a score between 0 and 1 means it occurs in both to some degree. For a new, unlabelled document, we can then calculate the average of all the term scores and predict a label based on whether it is above or below 0.5. This process is illustrated in Table 11.3.

For each term, we calculate in what percentage of documents it occurs for each label, and then average those scores to get a final prediction. The term "domestication" occurs in 50% of Neolithic documents, and not at all in Bronze Age documents, meaning it is an indicator (or feature) of a document belonging to the Neolithic class. "flint" occurs in both classes but more in Neolithic and "knapping" occurs in both equally, meaning it does not indicate either class. Then finally, "microscope" does not occur in either class, so also does not affect the classification.

Table 11.4 Simplified example of four documents, with term frequencies for the terms 'flint' and 'bronze'

Document number	Document content	Flint TF	Bronze TF
1	Flint bronze flint bronze flint	3	2
2	Flint flint flint flint	4	0
3	Bronze flint bronze bronze bronze	1	4
4	Bronze bronze bronze	0	3

When the scores are averaged, we can see that the label "Neolithic" is predicted with 66.6% certainty. Of course, this is a very simplified model of classification, but should give an insight into how Machine Learning algorithms deal with text data. In real world examples, there are often many more possible labels, many more terms to take into account, and possibly bias due to differences in document size, all of which complicate matters.

Unsupervised learning does not use any labelled data, but can still make subdivisions in data. In essence, most unsupervised learning methods are some variation of a clustering algorithm. Of course, archaeologists are very familiar with clustering algorithms, and we have been using these methods for at least 40 years (Doran & Hodson, 1975). Some examples include geospatial clustering of finds (Bogdanovic, 2015) and clustering artefacts into a typology (Gilboa et al., 2004). It is possible to do the same with text data, after transforming the text into a vector (as discussed in Sect. 11.4.6). The most used unsupervised learning technique used for archaeological texts is topic modelling: automatically creating a number of clusters, each with a certain topic, defined by which words are most frequent in that cluster.

A simplified example is provided in Table 11.4, where four documents are preprocessed to only contain the terms 'flint' and 'bronze', and the term frequencies are shown for each. At this point, the documents have been vectorised: for each document there is a vector with two dimensions (the dimensions being flint and bronze). This can also be expressed as a list of vectors (here displayed in Python syntax):

```
{
    1 : [3 , 2],
    2 : [4 , 0],
    3 : [1 , 4],
    4 : [0 , 3]
}
```

For each document number, there is a corresponding list containing two numbers (a two-dimensional vector). By treating these numbers as x and y values, we can easily plot this as a scatter plot to visualise the data: see Fig. 11.2. Here, the document vectors are plotted in two-dimensional vector space, and an algorithm has been applied to cluster the points into two groups based on their position in the plot. In essence, this is how clustering text data works, although normally the vectors used have more than two dimensions, often hundreds or even thousands, which makes

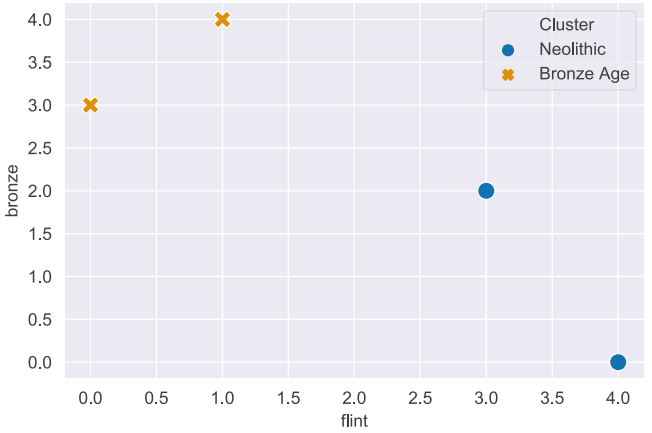

Fig. 11.2 Scatter plot of the data from Table 11.4. Points have been clustered and assigned a label and colour

these hyper-dimensional vector spaces difficult to intuitively illustrate.[7] The group label and colour have been manually assigned, and this is an important point: any clustering algorithm will return a number of clusters, but it can not assign a label, this has to be done manually afterwards by inspecting the data.

In this example, it was very easy to assign topic labels, as there are two well-defined groups with different content, but this is not always the case. An example is the work by (Plets et al., 2021), who used topic modelling to try and detect changes in theoretical thought in archaeology over time. Unfortunately, the clusters presented by the algorithm could not be assigned to different schools of thought. Another problem with (some) clustering algorithms is that they are non-deterministic, i.e. running the same analysis on the same data with the same settings will produce differing results every time. The size of the difference can be small or substantial, and any conclusion based on the method will have to take this into account.

As unsupervised learning does not provide actual labels for our data, it is not often used. Therefore, the rest of this chapter will mainly focus on the characteristics of supervised learning.

[7] There are possibilities to display multi-dimensional data in two or three dimensions: an often used method is Principal Component Analysis (Wold et al., 1987) which 'flattens' data, but also loses complexity.

11.5.2 Training Data and Validation

For any supervised Machine Learning method, training data with annotated labels is required for the algorithm to learn from. This training data is sometimes also called the 'ground truth'. Depending on the task, different types of labels are needed. In the case of document classification, one or more labels is needed for each document. For Named Entity Recognition, a label is required for each token. Such a combination of an observation (a document or a token) and a corresponding label is called a sample. It is important that the training data is representative of the entire data set, so the algorithm can learn—and deal with—the variety that exists in the data.

Once the labelled data has been created, it is required to split the data into a train set and a test set. The train set is used to train the model, so the algorithm uses these samples to create statistical relations. The test set is then used to evaluate how well the model is performing. This is done by letting the model predict labels on the test set, and then comparing them to the ground truth labels to calculate a performance metric (see Sect. 11.5.4). It is important that the model does not 'see' the test set during training, as that would give an unfair advantage, and the performance score would not reflect the effectiveness it will have on unlabelled data.

Often, the data is split into 80% train set and 20% test set, also called an 80/20 split. But other splits with more or less test data can be useful, depending on the task and the amount of data available. However, such a static split does come with a caveat: if the test set coincidentally happens to be very easy or hard to predict, this does not truly reflect how well the model would perform on new data. To prevent this, it is often better to perform leave-one-out cross validation. This means that the data is split into k equal sized chunks, and the model is trained k times, each time using one of the chunks as the test set, and the rest of the chunks (k-1) as training data. Afterwards, the performance metrics are averaged across the k runs to provide a more well-rounded indication of the model's quality.

For any task, a relatively large number of samples is needed for the algorithm to be effective. Unfortunately, there is no predefined number of samples which would guarantee good performance: each task is different and has varying levels of complexity, which influences the amount of data needed. For some simpler tasks with just two possible labels (a binary task), 300 to 500 samples might be enough, but for e.g. complex NER, thousands of examples are needed for each target entity. One way to determine if more data will improve the performance, is by again splitting the data into k chunks (with k often being 10), and running the algorithm k-1 times, starting with 2 chunks (1 train, 1 test) and every time adding one chunk of data (which becomes the test set). The performance scores can then be plotted in a line graph to judge whether adding more labelled data would help. A curve that flattens out means adding more data will probably not help, but a curve that has not flattened out yet indicates more data will probably increase the performance (Brandsen et al., 2020).

11.5.3 Commonly Used Algorithms for Information Extraction

Many algorithms have been developed for Machine Learning, each with different strengths and weaknesses. To give an idea of which are useful for Information Extraction, some commonly used methods are discussed here for each type of task. This list is far from exhaustive, for a more complete list see (Mohri et al., 2018).

For text classification, the most commonly used algorithm used to be Naive Bayes (NB), which uses the probabilities of known events to predict new events. In fact, the example in Table 11.3 is a form of NB. It is particularly useful when working with small training data, as it learns quickly compared to other methods which require more data. However, this method is not very powerful or good at handling complex data, and has been largely superseded by the Support Vector Machines (SVM) algorithm (Cortes & Vapnik, 1995).

SVM works by plotting vectors in a space (like in Fig. 11.2), and drawing a line (called a hyperplane in multidimensional space) dividing the points so the distance between all points and the hyperplane is maximised. Any new vectors will be assigned a label depending on which side of the hyperplane it is plotted. To illustrate this, a hyperplane has been added in Fig. 11.3, and a new, unlabelled point is added (green square). The red point—based on this hyperplane—would be classified as 'Neolithic' by the SVM. In reality, these hyperplanes are never straight, but bend around the vector points in hyper-dimensional space. This can be calculated, but unfortunately not visualised.

For NB and SVM, the order of the samples does not matter and is not taken into account. However, for Named Entity Recognition, a lot of information is encoded in the order of—and context around—a token. Think of e.g. the time period entity,

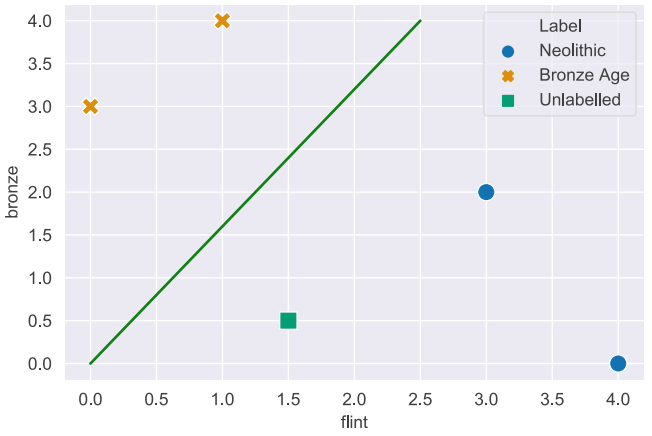

Fig. 11.3 Scatter plot of the data from Table 11.4. Points have been clustered and assigned a label and colour. A hyperplane dividing the two groups of points has been added in green. A new, unlabelled vector is displayed (green square)

it is very likely that time periods are preceded by the tokens "around" or "from", for example "we found a house from 1800 BCE". Having information about tokens before and after the current token the algorithm is trying to label is very useful, and so for NER, other algorithms are more effective. The most well-known one is the Conditional Random Fields (CRF) algorithm, and is generally the starting point for any sequence classification problem. (Joachims, 1998). It is relatively easy to use, does not require much computing power or time to run, and it generally produces good results. NB, SVM and CRF are all available to use via the `scikit-learn` Python library (Pedregosa et al., 2011), among others.

For both document classification and NER, neural networks (also known as Deep Learning) have seen an increase in popularity over the last decade. As they are able to capture complexity more accurately than 'traditional' algorithms, they can provide state-of-the-art performance. For document classification, Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) are often used. For NER, the Bidirectional Long Short Term Memory (BiLSTM) algorithm is popular, as well as the Bidirectional Encoder Representations from Transformers (BERT) architecture. BERT is discussed in more detail in Sect. 11.5.6 below.

Clustering is often performed using the k-means algorithm. Also used extensively with other types of data, k-means aims to group vectors into k clusters by minimising the within-cluster variance. Specifically for topic modelling, LDA (Latent Dirichlet Allocation) is often used, which can relatively easily be implemented with the pyLDAvis Python library (Sievert & Shirley, 2014). In Fig. 11.4 an example is shown of the output of pyLDAvis, displaying ten clusters of documents about ancient fire use. Topic number 8 has been highlighted, and the top relevant terms for that topic displayed on the right. Judging from the top terms, this particular cluster seems to be about burning bones and/or cremations.

11.5.4 Evaluation and Performance Metrics

To see how well an algorithm performs, we need to evaluate the output. For unsupervised learning, there are no target labels, so it is not possible to do a quantitative evaluation, and a qualitative evaluation is needed by manually inspecting the outcome. For supervised learning, it is possible to quantitatively measure the output, as we can compare the labels predicted by the algorithm to the true labels assigned by human annotators, and calculate performance metrics. However, due to the fuzziness and ambiguity in archaeological data, sometimes a manual inspection of the predicted labels is warranted, to see in detail where the algorithm is correct and incorrect (or nearly correct). However, a performance metric should always be calculated when possible, as this gives an overview of the performance over the entire test set, but also because this makes it possible to compare different methods on the same data, and promotes reproducible open science.

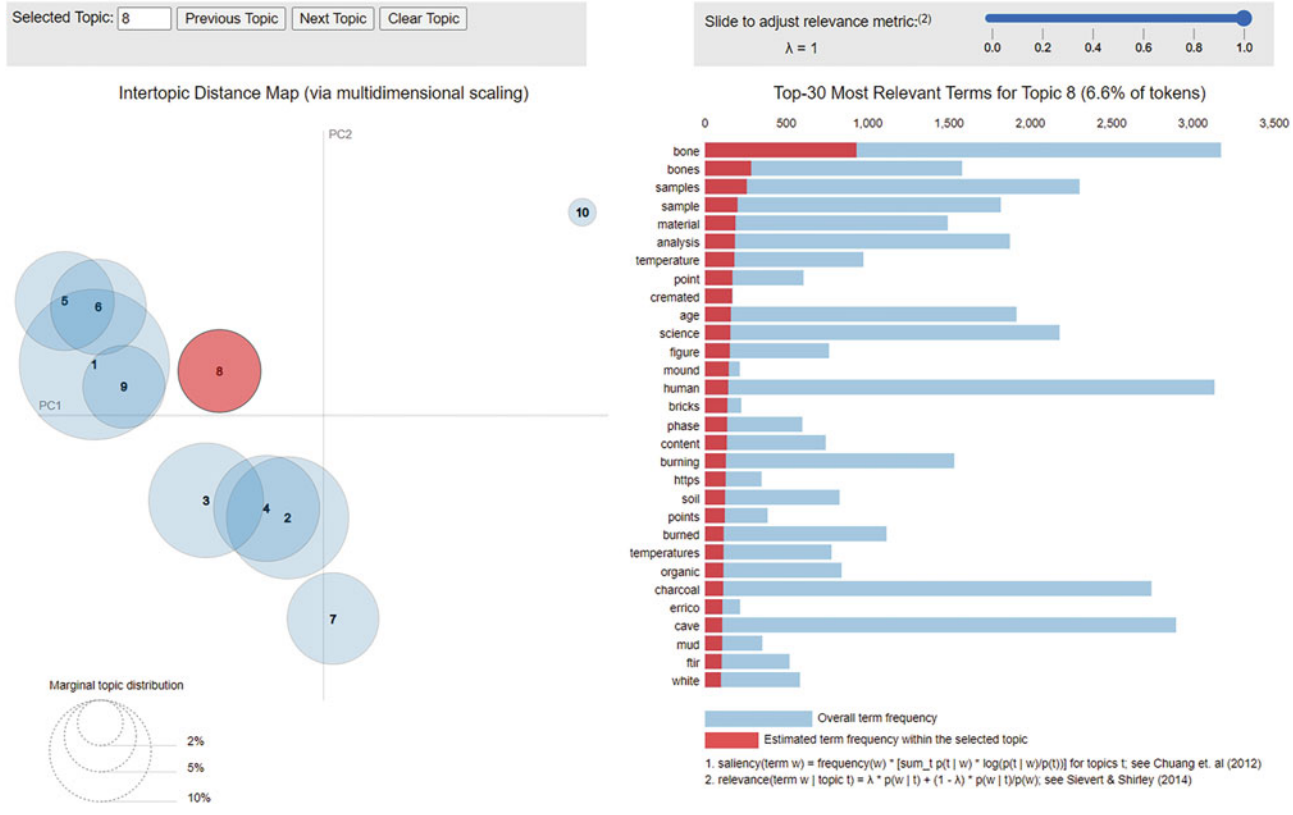

Fig. 11.4 Topic model visualisation by pyLDAvis

Table 11.5 Illustrating the true/false positive/negative categories

		Prediction	
		True	False
Label	True	tp	fn
	False	fp	tn

In the rest of this section, the most common metrics for text mining are discussed, but many more exist, and it is worth investigating which one is most suitable for a given task. Most metrics involve calculations of percentages between correctly and incorrectly classified items. A label is predicted by the algorithm for each item in the test set, and those predicted labels are compared to the true labels. Each prediction can then be assigned to one of the categories listed below. The categories and metrics are further explained with an archaeological example: imagine a document classification task where the goal is to automatically label a large set of papers as being *relevant* or *irrelevant* to a certain research topic, e.g. Early Medieval cremations in Europe. As the amount of possibly relevant papers is too large to manually inspect, using a Machine Learning algorithm to make a preselection could be useful.

- True positive (tp). When a paper is relevant, and the label is correctly predicted as 'relevant'.
- True negative (tn). When a paper is irrelevant, and the label is correctly predicted as 'irrelevant'.
- False negative (fn). When a paper is relevant, but the label is incorrectly predicted as 'irrelevant'. More simply put: a paper that has not been recognised as relevant by the system.
- False positive (fp). When a paper is not relevant, but the label is incorrectly predicted as 'relevant'. More simply put: the system thinks a paper is relevant when it is not.

These categories are further illustrated in Table 11.5. Once all the items have been assigned a category, it is possible to calculate performance metrics. The most used measures in Machine Learning in general are recall, precision and F1 score..

Recall shows what proportion of actual positives was identified correctly. For our example, it indicates out of all the relevant papers, what percentage have been correctly labelled as 'relevant'. It can also be viewed as the percentage of papers that have been found. It is defined as follows:

$$\text{Recall} = \frac{tp}{tp + fn} \tag{11.1}$$

Precision shows what proportion of positive identifications was actually correct. For our example, it indicates out of all the papers labelled as 'relevant', what percentage was actually relevant. In essence, this means that it shows that when an algorithm predicts a label, how often it is right. It is defined as follows:

$$\text{Precision} = \frac{tp}{tp + fp} \tag{11.2}$$

The F1 score (or F measure) combines recall and precision to provide an overall evaluation metric. More specifically, it is the harmonic mean of precision and recall, and is defined as:

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{11.3}$$

The 1 in F1 means that recall and precision are equally important (and thus equally weighted) when calculating the harmonic mean. But in some cases, either recall or precision are more important, in which case the F score can be weighted to favour recall or precision more. This is done by changing to the F0.5 score (precision is 2 times more important/weighted) or F2 score (recall is 2 times more important/weighted). For example, (Brandsen et al., 2019) shows that when searching for documents, Dutch archaeologists are more interested in getting as many relevant documents as possible, even if this means getting more irrelevant documents. This means that the recall is more important, and the F2 score would be more suited for that task.

Other metrics are less popular, but can be useful in certain situations. These include the ROC (Receiver Operating Characteristic) curve, the related AUC (Area Under the ROC Curve), and the MCC (Matthews Correlation Coefficient) (Verschoof-van der Vaart & Landauer, 2021). If a less popular metric is chosen, it is useful to also include the most common metric(s) for the task as well, to be able to compare algorithms between studies.

Generally, these metrics are not calculated manually. Most Machine Learning libraries will have functions that can automatically calculate the metrics, based on an input of predicted labels and correct labels. For Python, the Metrics functions of the `scikit-learn` library have the metrics discussed here available, among many others.

11.5.5 Word Embeddings

Word embeddings are a different way to represent tokens. Instead of using the actual string (or a number assigned to that string), word embedding algorithms

convert tokens into vectors. Instead of creating a vector for each document (like in Sect. 11.4.6), a vector is created for each token in the document. The vectors are created by the word embeddings algorithm in such a way that words which occur in similar contexts (i.e. have similar words near it in sentences), have similar vectors (i.e. are near each other in the vector space). This is based on the distributional hypothesis: words that have similar contexts will have similar meanings (Harris, 1954). Once the vectors for the individual tokens have been calculated, a single vector for the document can be created (for example by averaging all the token vectors).

Word embeddings are useful because to a computer, "axe" and "adze" are two completely unrelated strings, the computer does not know they are semantically similar. However, if the vectors of these two words are near each other, the computer can use this information to understand that they are similar. To illustrate this, the following two sentences (before and after preprocessing) would have substantially different vectors when using the method from Sect. 11.4.6:

- "The axe was used to chop wood" (axe used chop wood)
- "The birch was carved with an adze" (birch carved adze)

In fact, after preprocessing, none of the tokens overlap between sentences. Yet it is clear to humans that these sentences have substantial semantic similarity. Assuming the word embeddings have been created correctly, these two sentences would be quite similar: axe is similar to adze, carve is similar to chop, and birch is similar to wood. And when averaged into a document vector, the computer understands these sentences to be similar, even though the tokens are completely different. This makes word embeddings incredibly powerful for dealing with text data, and consequently, it has been applied with great success to many tasks: from document classification and NER, to automatically expanding search queries and tracking the change of meanings of words over time.

Word embeddings can be created by multiple algorithms, the most popular currently are word2vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2016). Instead of just averaging word vectors to get to a document vector, it is also possible to use the doc2vec model (Le & Mikolov, 2014) to create more sophisticated document vectors. All these models can be implemented in Python using the `gensim` library (Rehurek & Sojka, 2010).

11.5.6 Transfer Learning

While word embeddings are a significant improvement over the bag-of-words model, the current state of the art in NLP is transfer learning, specifically transformer based methods. These Deep Learning algorithms can 'learn' language from extremely large unlabelled text collections (billions of tokens) to create a language model, and can then use this model to better perform specific tasks. The idea behind these language models, is that they mimic human behaviour: by already knowing a language, it is easier to try and predict classes. The most well-

known architecture is called BERT (Bidirectional Encoder Representations from Transformers), developed by researchers at Google (Devlin et al., 2019).

Similar to word embeddings, BERT also creates vectors for tokens. However, traditional word embeddings such as word2vec are context independent, meaning a token will always have the same vector, regardless of its context. In this case, the word 'flint' will have the same vector in "a flint axe" and "Mr. Flint" while being semantically very different. BERT produces context-dependent embeddings, meaning the vector of a token is different if it occurs in a different context. This means that BERT is particularly useful for tasks where synonymy and polysemy are a problem, and can handle more complex tasks with higher performance.

While BERT is being used extensively in other domains for a wide range of tasks, in archaeology it has not been used much yet. An exception to this is the work by Brandsen et al. (2021), who used BERT for NER, showing substantial improvement over a CRF based method.

BERT does have large potential for use in archaeology, as it leverages *unlabelled* data to train the neural net. Normally for deep learning algorithms, a very large amount of labelled data is needed to train the network, which often is not available in our domain. By creating a language model with unlabelled data, only a modest amount of labelled data is needed to fine-tune the model on a specific NLP task. The unlabelled training data does not necessarily need to be archaeological data either, as long as it is in the same language, hence why it is called 'transfer learning', transferring knowledge from one domain to another. However, research does show that using unlabelled training data from the domain itself can lead to modest increases in performance (Lee et al., 2019; Beltagy et al., 2020; Brandsen et al., 2021).

11.6 Conclusions

This chapter has described various Information Extraction methods, how to perform these using Machine Learning, and given an introduction to data preprocessing and the evaluation of text mining algorithms, with a focus on practical archaeological examples. This provides a snapshot of the current state of research, as well as some ideas and inspiration for future directions.

Even though a large proportion of this chapter is dedicated to machine automation, computers are not going to replace archaeologists any time soon, as also noted by other archaeologists working with Machine Learning (Verschoof-van der Vaart et al., 2020; Traviglia et al., 2016). While computers are great at calculating answers, they are not able to ask any questions: formulating research ideas and analysing the output of algorithms will still have to be done by humans. A certain level of creativity and ability to 'connect the dots' is needed in science, which we need human brains for. While neural networks are getting increasingly complex and are starting to mimic human learning, they are still rudimentary when compared to the incredible ability of humans to learn from scratch, connect ideas, and think out of

the box, while algorithms are (quite literally) bound by their 'box', or the limits of the programming that created them.

Instead, computational tools are meant to further enhance the archaeologist's ability to draw meaningful conclusions from raw data and to make this process more efficient. Outsourcing menial tasks to e.g. students and volunteers has a long history in archaeology, and science as a whole. The more we can replace this valuable human time with relatively unvaluable computing time, the more we can focus on the interesting parts of archaeology: drawing conclusions and building theories relating to past human behaviour.

However, this new big data paradigm (Löwenborg, 2018) and the associated techniques also pose new challenges (Kintigh et al., 2014; Gattiglia, 2015). An example is the reliability of data. While data has always been central to archaeological knowledge, in this new paradigm large data sets can be presumed to be unproblematic, and any problems with quality or reliability to be overcome purely by the quantity of data (Huggett, 2020). This can cause the conceptual understanding of the creation of archaeological data—gained over decades of discussion—to be overlooked when performing these large scale syntheses (Cunningham & MacEachern, 2016). At the same time, discussions around big data have seen a renewed interest in the relation between data, and the knowledge created from this data (Leonelli, 2015).

As big data is getting increasingly ubiquitous in archaeology, it seems inevitable that computational methods to find, combine, and analyse nuggets of information from large data sets will become increasingly common place. As other domains—and specifically computer science scholars—push the state of the art of NLP towards ever-increasing performance, we as archaeologists can use and adapt these new tools with relative ease, or collaborate with experts. Using these methods and applying them to our own data, for our own research questions, we can perform better synthesising research at larger scales, leading to a better, more thorough understanding of the past.

References

Amrani, A., Abajian, V., & Kodratoff, Y. (2008). A chain of text-mining to extract information in archaeology. In *Information and communication technologies: From theory to applications, ICTTA 2008, Damascus, Syria* (pp. 1–5). https://doi.org/10.1109/ICTTA.2008.4529905

Beel, J., Gipp, B., Langer, S., & Breitinger, C. (2016). Research-paper recommender systems: A literature survey. *International Journal on Digital Libraries, 17*(4), 305–338. https://doi.org/10.1007/S00799-015-0156-0

Beltagy, I., Lo, K., & Cohan, A. (2020). SCIBERT: A pretrained language model for scientific text. In *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*. Hong Kong: Association for Computational Linguistics. https://doi.org/10.18653/v1/d19-1371

Bevan, A. (2015). The data deluge. *Antiquity 89*(348), 1473–1484. https://doi.org/10.15184/aqy.2015.102

Bickler, S. H. (2021). Machine learning arrives in archaeology. *Advances in Archaeological Practice, 9*(2), 186–191. https://doi.org/10.1017/aap.2021.6

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with Python*. Sebastopol: O'Reilly.

Bogdanovic, I. (2015). Spatial cluster detection in archaeology: Current theory and practice. In *Mathematics and archaeology* (pp. pp 366–382). Boca Raton: CRC Press.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics, 5*(1), 135–146.

Brandsen, A., & Koole, M. (2021). Labelling the past: Data set creation and multi-label classification of dutch archaeological excavation reports. *Language Resources and Evaluation, 56*, 543–572. https://doi.org/10.1007/s10579-021-09552-6

Brandsen, A., Lambers, K., Verberne, S., & Wansleeben, M. (2019). User requirement solicitation for an information retrieval system applied to Dutch grey literature in the archaeology domain. *Journal of Computer Applications in Archaeology, 2*(1):21–30, https://doi.org/10.5334/jcaa.33

Brandsen, A., & Lippok, F. (2021). A burning question – Using an intelligent grey literature search engine to change our views on early medieval burial practices in the Netherlands. *Journal of Archaeological Science, 133*, 105456. https://doi.org/10.1016/j.jas.2021.105456

Brandsen, A., Verberne, S., Lambers, K., & Wansleeben, M. (2021). Can BERT dig it? - Named entity recognition for information retrieval in the archaeology domain. http://arxiv.org/abs/2106.07742

Brandsen, A., Verberne, S., Wansleeben, M., & Lambers, K. (2020). Creating a dataset for named entity recognition in the archaeology domain. In *Proceedings of the 12th Language Resources and Evaluation Conference* (pp. 4573–4577). Marseille: European Language Resources Association. https://www.aclweb.org/anthology/2020.lrec-1.562/

Brandt, R., Drenth, E., Montforts, M., Proos, R., Roorda, I., & Wiemer, R. (1992). Archeologisch Basisregister. Tech. Rep., Rijksdienst voor Cultureel Erfgoed, Amersfoort.

Byrne, K., & Klein, E. (2010). Automatic extraction of archaeological events from text. In B. Frischer, J. Crawford, & D. Koller (Eds.), *Making history interactive: Computer applications and quantitative methods in archaeology 2009*. BAR International Series (vol. 2079, pp. pp 48–56). Oxford.

Chowdhury, G. G. (2005). Natural language processing. *Annual Review of Information Science and Technology, 37*(1), 51–89. https://doi.org/10.1002/aris.1440370103

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297. https://doi.org/10.1007/BF00994018

Cunliffe, E., & Curini, L. (2018). ISIS and heritage destruction: A sentiment analysis. *Antiquity, 92*(364), 1094–1111. https://doi.org/10.15184/AQY.2018.134

Cunningham, J. J., & MacEachern, S. (2016). Ethnoarchaeology as slow science. *World Archaeology, 48*(5), 628–641.

Davis, D. S. (2020). Defining what we study: The contribution of machine automation in archaeological research. *Digital Applications in Archaeology and Cultural Heritage, 18*, e00152. https://doi.org/10.1016/J.DAACH.2020.E00152

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (vol. 1, pp. 4171–4186). Minnesota: Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423

Doran, J., & Hodson, F. (1975). *Mathematics and computers in archaeology*. Harvard: Harvard University Press.

Feldman, R., & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge: Cambridge University Press.

Fiorucci, M., Khoroshiltseva, M., Pontil, M., Traviglia, A., Del Bue, A., & James, S. (2020). Machine learning for cultural heritage: A survey. *Pattern Recognition Letters, 133*, 102–108. https://doi.org/10.1016/j.patrec.2020.02.017

Fischer, A., Londen, H. V., Bercken, A. B. V. D., Visser, R., & Renes, J. (2021). NAR 68 Urban farming and ruralisation in the Netherlands (1250 up to the nineteenth century), unravelling farming practice and the use of (open) space by synthesising archaeological reports using text mining. Nederlandse Archeologische Rapporten (NAR) 68.

Gattiglia, G. (2015). Think big about data: Archaeology and the big data challenge. *Archäologische Informationen, 38*(1), 113–124. https://doi.org/10.11588/ai.2015.1.26155

Gibbs, M., & Colley, S. (2012). Digital preservation: Online access and historical archaeology 'grey literature' from New South Wales, Australia. *Australian Archaeology, 75*, 95–103. https://doi.org/10.1080/03122417.2012.11681957

Gilboa, A., Karasik, A., Sharon, I., & Smilansky, U. (2004). Towards computerized typology and classification of ceramics. *Journal of Archaeological Science, 31*(6), 681–694. https://doi.org/10.1016/j.jas.2003.10.013

Gilman, P., & Newman, M. (2007). Informing the future of the past: Guidelines for historic environment records (2nd edn.). Tech. Rep., ADS, ALGAO UK, English Heritage, Historic Scotland, RCAHMS and RCAHMW.

Grove, M., & Blinkhorn, J. (2020). Neural networks differentiate between middle and later stone age lithic assemblages in eastern Africa. *PloS One, 15*(8), e0237528.

Gualandi, M. L., Gattiglia, G., & Anichini, F. (2021). An open system for collection and automatic recognition of pottery through neural network algorithms. *Heritage, 4*(1), 140–159.

Harris, Z. S. (1954). Distributional structure. *Word, 10*(2–3), 146–162.

Huggett, J. (2020). Is big digital data different? Towards a new archaeological paradigm. *Journal of Field Archaeology, 45*(suppl. 1), S8–S17. https://doi.org/10.1080/00934690.2020.1713281

International Committee for Documentation (CIDOC). (2014). Information and documentation - A reference ontology for the interchange of cultural heritage information (ISO Standard No. 21127:2014). Tech. Rep., International Organization for Standardization. https://www.iso.org/standard/57832.html

Jackson, S., Richissin, C. E., McCabe, E. E., & Lee, J. J. (2020). Data-informed tools for archaeological reflexivity: Examining the substance of bone through a meta-analysis of academic texts. *Internet Archaeology, 55*. https://doi.org/10.11141/ia.55.12

Jeffrey, S., Richards, J., Ciravegna, F., Waller, S., Chapman, S., & Zhang, Z. (2009). The Archaeotools project: Faceted classification and natural language processing in an archaeological context. *Philosophical Transactions Series A, Mathematical, Physical, and Engineering Sciences, 367*(1897), 2507–19. https://doi.org/10.1098/rsta.2009.0038

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In: *Machine learning: ECML-98* (pp. 137–142). Berlin: Springer.

Kintigh, K. W., Altschul, J. H., Beaudry, M. C., Drennan, R. D., Kinzig, A. P., Kohler, T. A., Limp, W. F., Maschner, H. D., Michener, W. K., Pauketat, T. R., Peregrine, P., Sabloff, J. A., Wilkinson, T. J., Wright, H. T., & Zeder, M. A. (2014). Grand challenges for archaeology. *American Antiquity, 79*(1), 5–24.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *International Conference on Machine Learning*. Proceedings of Machine Learning Research (pp. 1188–1196).

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2019). BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics, 36*(4), 1234–1240. https://doi.org/10.1093/bioinformatics/btz682

Leonelli, S. (2015). What counts as scientific data? A relational framework. *Philosophy of Science, 82*(5), 810–821.

Löwenborg, D. (2018). Knowledge production with data from archaeological excavations. In *Archaeology and archaeological information in the digital society* (pp. 37–53). Milton Park: Routledge.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. Cambridge: Cambridge University Press.

Manning, C. D., Ragahvan, P., & Schutze, H. (2009). *An introduction to information retrieval*. Cambridge: Cambridge University Press. https://doi.org/10.1109/LPT.2009.2020494

Merali, Z., & Smith, J. (1985). *Optical character recognition: The technology and its application in information units and libraries*. Wetherby: Boston Spa.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*.

Mitchell, T. (1997). *Machine learning*. New York: McGraw Hill.

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). *Foundations of machine learning* (2nd edn.). Cambridge: MIT Press.

Nash, B. S., & Prewitt, E. R. (2016). The use of artificial neural networks in projectile point typology. *Lithic Technology, 41*(3), 194–211.

Niccolucci, F., & Richards, J. D. (2013). ARIADNE: Advanced research infrastructures for archaeological dataset networking in Europe. *International Journal of Humanities and Arts Computing, 7*(1–2), 70–88. https://doi.org/10.3366/ijhac.2013.0082

Paijmans, H., & Brandsen, A. (2010). Searching in archaeological texts: Problems and solutions using an artificial intelligence approach. *PalArch's Journal of Archaeology of Egypt/Egyptology, 7*(2), 1–6.

Paolanti, M., Pierdicca, R., Martini, M., Felicetti, A., Malinverni, E., Frontoni, E., & Zingaretti, P. (2019). Deep convolutional neural networks for sentiment analysis of cultural heritage. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 4215*, 871–878.

Pawlowicz, L. M., & Downum, C. E. (2021). Applications of deep learning to decorated ceramic typology and classification: A case study using Tusayan White Ware from Northeast Arizona. *Journal of Archaeological Science, 130*, 105375. https://doi.org/10.1016/j.jas.2021.105375

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12*, 2825–2830.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543)

Plets, G., Huijnen, P., & van Oeveren, D. (2021). Excavating archaeological texts: Applying digital humanities to the study of archaeological thought and banal nationalism. *Journal of Field Archaeology, 46*, 289–302. https://doi.org/10.1080/00934690.2021.1899889

Porter, M. F. (1980). An algorithm for suffix stripping. *Program, 14*(3), 130–137.

Rehurek, R., & Sojka, P. (2010). Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta: ELRA.

Richards, J., Tudhope, D., & Vlachidis, A. (2015). Text mining in archaeology: Extracting information from archaeological reports. In J. A. Barcelo & I. Bogdanovic (Eds.), *Mathematics and archaeology* (pp. 240–254). Boca Raton: CRC Press. https://doi.org/10.1201/b18530-15

Riley, M. D. (1989). Some applications of tree-based modelling to speech and language. In *Proceedings of the Workshop on Speech and Natural Language, Association for Computational Linguistics (ACL)* (pp. 339–352). https://doi.org/10.3115/1075434.1075492

Sanders, D. H. (2018). Neural networks, AI, phone-based VR, machine learning, computer vision and the CUNAT automated translation app–not your father's archaeological toolkit. In *2018 3rd Digital Heritage International Congress (DigitalHERITAGE) Held Jointly with 2018 24th International Conference on Virtual Systems & Multimedia (VSMM 2018)* (pp. 1–5). Piscataway: IEEE.

Sievert, C., & Shirley, K. (2014). LDAvis: A method for visualizing and interpreting topics. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, pp. 63–70

Sommerschield, T. (2020). Ralegh radford Rome awards: Restoring ancient text using machine learning: A case-study on Greek and Latin epigraphy. *Papers of the British School at Rome, 88*, 387–388. https://doi.org/10.1017/S0068246220000240

Talboom, L. (2017). Improving the discoverability of zooarchaeological data with the help of Natural Language Processing. Master's thesis, University of York.

Talks, A. (2019). An exploration of NLP and NER for enhanced search in osteoarchaeological and palaeopathological textual resources. Master's Thesis, University of York.

Tjong Kim Sang, E. F. (2002). Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*.

Traviglia, A., Cowley, D., & Lambers, K. (2016). Finding common ground: Human and computer vision in archaeological prospection. *AARGnews-The Newsletter of the Aerial Archaeology Research Group, 53*, 11–24.

Trier, Ø. D., Salberg, A. B., & Pilø, L. H. (2018). Semi-automatic mapping of charcoal kilns from airborne laser scanning data using deep learning. In *CAA2016: Oceans of Data. Proceedings of the 44th Conference on Computer Applications and Quantitative Methods in Archaeology* (pp. 219–231). Oxford: Archaeopress.

Tudhope, D., May, K., Binding, C., & Vlachidis, A. (2011). Connecting archaeological data and grey literature via semantic cross search. *Internet Archaeology, 30*(30). https://doi.org/10.11141/ia.30.5

Turney, P. (2002). Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp 417–424

Verschoof-van der Vaart, W. B., Lambers, K., Kowalczyk, W., & Bourgeois, Q. P. (2020). Combining deep learning and location-based ranking for large-scale archaeological prospection of LiDAR data from the Netherlands. *ISPRS International Journal of Geo-Information, 9*(5), 293. https://doi.org/10.3390/ijgi9050293

Verschoof-van der Vaart, W. B., & Landauer, J. (2021). Using CarcassonNet to automatically detect and trace hollow roads in LiDAR data from the Netherlands. *Journal of Cultural Heritage, 47*, 143–154. https://doi.org/10.1016/j.culher.2020.10.009

Vince, A. (1996). Editorial. *Internet Archaeology, 1*. https://doi.org/10.11141/ia.1.7

Vlachidis, A. (2012). *Semantic indexing via knowledge organization systems: Applying the CIDOC-CRM to archaeological grey literature*. Unpublished PhD Thesis, University of South Wales (USW).

Vlachidis, A., Tudhope, D., & Wansleeben, M. (2021). Knowledge-based named entity recognition of archaeological concepts in Dutch. In E. Garoufallou & M. A. Ovalle-Perandones (Eds.), *14th International Conference on Metadata and Semantic Research* (pp. 53–64). Cham: Springer. https://doi.org/10.1007/978-3-030-71903-6_6

Vlachidis, A., Tudhope, D., Wansleeben, M., Azzopardi, J., Green, K., Xia, L., & Wright, H. (2017). D16.4: Final report on natural language processing. Tech. Rep., ARIADNE. http://legacy.ariadne-infrastructure.eu/wp-content/uploads/2019/01/D16.4_Final_Report_on_Natural_Language_Processing_Final.pdf

Wilcke, W. X., de Boer, V., de Kleijn, M. T., van Harmelen, F. A., & Scholten, H. J. (2019). User-centric pattern mining on knowledge graphs: An archaeological case study. *Journal of Web Semantics, 59*, 1–10. https://doi.org/10.1016/j.websem.2018.12.004

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J. W., da Silva Santos, L. B., Bourne, P. E., Bouwman, J., Brookes, A. J., Clark, T., Crosas, M., Dillo, I., Dumon, O., Edmunds, S., Evelo, C. T., Finkers, R., ... Mons, B. (2016) The FAIR guiding principles for scientific data management and stewardship. *Scientific Data, 3*, 160018. https://doi.org/10.1038/sdata.2016.18

Wold, S., Esbensen, K., & Geladi, P. (1987). Principal component analysis. *Chemometrics and Intelligent Laboratory Systems, 2*(1–3), 37–52.