Human-centric Approaches to Annotation for Classification and Named Entity Recognition Tasks

Alexander Downing

University of the West of England

15028101

UFCFR4-45-3

Computing Project

Contents

Abstract

Annotation projects are gruelling but essential components of typical Machine Learning (ML) workflows. This is particularly true within the subfield of Natural Language Processing (NLP), where large amounts of amorphous, free-form text require rigorous analysis and structuring before use. This project presents the Classification Annotation System in Python (CL-ASP) as the basis of a study into ways of expediting the annotation of text, for use in supervised learning environments. Interactive visualisations are explored and developed within the system. These elements are derived using an ensemble of approaches, encapsulating unsupervised ML methods in support of corpus investigation, and Active Learning (AL) procedures to aid sample selection. Despite not realising the full system, CL-ASP currently shows promise in expanding upon the main research premise. However, in its current state cannot facilitate an empirical study required to assess the validity of its benefits against similar products or tools.

Acknowledgements

I would like to take the opportunity to thank my supervisor Dr. Phil Legg, for supporting me throughout my academic career, offering valuable insight and guidance towards this project, and giving me the space to produce my best work. Additionally, I would like to express my gratitude to my partner, Francesca, for her support and encouragement during my degree and beyond. Finally, I would like to express my deepest sympathies to those whose loved ones have sadly passed away during this time of crisis, and pay solemn tribute to their memory.

Introduction

Supervised Machine Learning (ML) techniques require a supply of training examples that are generally representative of the application domain. Therefore, it is understood that models cannot be reliably produced in this way without appropriately annotated datasets. Aside from data integration, the notion of data worthiness is a key issue facing some sectors of industry. Those with the ability to embrace information-intensive marketplace behaviours have become motivating examples due to their profitability, however, attempts to leverage material amassed in industrial silos can be challenging without proper consideration. Retrospectively poor standards of data management can typically result in material being suboptimal for supervised learning tasks. To rectify this, large-scale annotation projects must be undertaken. Generally, such a venture comes at great cost and effort, due to the arduousness of the task and sole reliance on human arbitration. Similarly, lack of investment in infrastructure and skills can equally diminish the means for industry to leverage its data. Often, the automation of a process requires a domain specialist to impart knowledge and guidance to substantiate modelling outcomes. This naturally adds to the total expense given the cruciality and/or scarcity of the expertise. Finally, it may be the case that commercial off-the-shelf (COTS) or opensource tools available to industry are unable to be generally applied, for example, the discontinuity in applying tools that draw from general language models in domains where deeply technical language is used. These scenarios exemplify a conventional starting point for most Natural Language Processing (NLP) tasks, where a set of correctly identified observations must be outlined before modelling can take place. For structured prediction problems, such as Named Entity Recognition (NER), these observations need to be recorded at the word-level, normally making the task of annotating a dataset for this purpose more gruelling. Conventional approaches to corpus annotation can force projects into restrictive waterfall procedures, halting experimentation in favour of upfront planning.

Additionally, tools or systems used to annotate corpora can also be prohibitively slow to use and lacking elements of interactivity or optimisation. For example, systems may allow users to provide annotations to preformed statements but offer no way of imposing knowledge aside from agreeing or disagreeing with what is given to them. Alternatively, the proposed method outlines the means to source and annotate both document and word-level annotations for a dataset by a single individual. The system allows users to explore and examine datasets, ascribe labels, and finally discharge an annotated corpus of text. Fundamentally, this project acknowledges a need to ameliorate current annotation methods and aims to explore novel ways to do so most effectively. Moreover, with the advances and democratisation of ML technology in recent years, it is reasonable to suggest that larger portions of industry will be encouraged to develop and integrate more ML components. Therefore, approaches to make annotation activities faster, more accessible to non-professionals, and more cost effective are all highly desirable objectives.

Questions

The primary and secondary questions this project proposes are as follows:

a. Are there more ways to expedite annotation using ML techniques? For example, the inclusion of interactive visualisation of the dataset derived from unsupervised methods.
b. Can this be combined with interactive or AL to further assist manual annotation projects?

Aims

The aim of the project is to develop and deliver a full system capable of optimising the annotation process of an unstructured dataset, reducing the overall effort of the human annotator. This should be achieved using some novel technique or approach, not applied in the current offering of NLP tools, theoretically building upon or improving the current state-of-the-art. These approaches will be derived from the field of visual analytics and artificial

intelligence. I will develop a modelling application, which incorporates a web-based user interface to support a variety of interactive views and instruments. Practically, end-users should be given the ability to visually inspect a given corpora of text and construct custom models to satisfy classification and NER tasks. Ideally, the complexity surrounding the modelling processes should be abstracted, simplified, and automated for the convenience of the user; with the graphical aspects serving as the sole means of interaction between a user the model. The procedure should be condensed into a system that utilises a user's visual intuitions, and innate understanding of language. Ultimately, the aim is to capitalise on these human traits, and translate a user's understanding into a model via the manipulation of various data projections, as opposed to outlining hand engineered linguistical rules. Thus, helping to impose the user's knowledge of language onto the statistically derived representations of the data, leveraging the strengths of both human and machine.

Objectives

1. Explore and analyse the literature regarding state-of-the-art annotation and visualisation tools/methods.
2. Identify a widely available dataset to form the basis of the annotation project. One ideally representative of a commercial environment, where the annotation assignment is realistic.
3. Outline an experimental case for the evaluation of the application against another tools/methods.
4. Create the application to demonstrate findings.
5. Evaluate the performance of the experiment.
6. Identify future work.

Grounds for declaring the project a success will be foremost whether an unstructured dataset can be manually annotated using the proposed system, to a standard where it is fit for use in NLP modelling activities. On a fundamental level, it needs to fulfil this purpose to be considered worthwhile. Secondarily, a comparison study between the proposed system

and another popular application could be undertaken, to gauge differences in usability, annotation speed and overall effectiveness. Similarly, how these two datasets would compare in terms of how they impact the outcome of a modelling activity could be interesting to experiment with.

Literature Review

Before being dispatched to a supervised ML algorithm, a body of text requires some form of supplementary metadata to allow for a mapping function to be generated. This metadata is introduced to help identify components or systems that contribute to human understanding, for example, marking aspects of grammar within the given text. These annotations must be accurate to provide meaningful results downstream, but also be applicable to the specific task being modelled. Typically, a single label denoting one of a finite set of categories is used for text classification tasks, however, multi-label variations of this task are also common.

Labelling schemes

NER tasks necessitate that labels be provided to every word found within the given text, each denoting the presence or absence of predefined entities to classify. Examples of such entities might include people, organisations, events, or similar. On occasion, an entity will span more than a single word, and so, is often a requirement that the labelling scheme used must also allow for the boundaries of an entity to be assigned. To this end, Ramshaw and Marcus (Ramshaw & Marcus, 1999) outlined the simple and widely adopted Inside-outside-beginning (IOB) scheme, to help signify entity boundaries. By conjoining the entity tag one of the following prefixes, here, *B* denotes the beginning of the entity, an *I* denotes the inside or continuation of a multi-token entity, and *O* the outside or otherwise superfluous non-entity tokens. More complex schemes, like the IOBES (*E* end of entity, *S* singleton entity) variant, include extra prefixes to the labelling scheme, facilitating additional granularity when defining entity boundaries. The connection between the use of more intricate labelling schemes and superior

modelling outcomes are often varied in literature. Some experimental results show no significant improvement over IOB, and instead, note the computational overhead and task complexity when introducing more tags. Others cite that more elaborate labelling schemes contribute considerably to yielding more accurate models (Ratinov & Roth, 2009) (Lample *et al.*, 2016) (Yang, Liang & Zhang, 2018).

Annotation tools

Many mature and highly developed tools are available for NLP projects, most with the ability to aid or automate a wide variety of text pre-processing and analysis tasks. Recent surveys of the state-of-the-art indicate that most newly developed tools are typically web applications, although stand-alone tools and plugins remained popular for some tasks or within some communities, for example, within the biomedical domains. The majority of tools under scrutiny supported the use of standard file formats (such as XML and JSON), were easy to install or included no requirement for installation, and included text span highlighting capabilities (Neves & Ševa, 2019). A subsection of the tools covered include some form of AL, human-in-the-loop interactive learning, or proactive learning to help expedite the annotation process by reducing human effort. The shortlist contains both commercial and non-commercial tools, among which, Prodigy (2020) and Tagtog (2020) stand out as products with similar functionality that this project aspires to produce, alongside a focus on reducing the number of people involved in an annotation task.

Prodigy, a recent offshoot of the creators of the open source NLP software library spaCy (spaCy, 2020), combines NLP functionality with an annotation interface. General-purpose pretrained models (built from GloVe (Pennington, Socher & Manning, 2014), and OntoNotes (Weischedel, *et al.*, 2013)) enable the NER capability, which can be trained to integrate custom entities via the annotation process. Hung *et al.* introduced Prodigy into their workflow whilst attempting to extract indicators of behavioural radicalisation from text. They describe that their most promising modelling

outcome was aided by their ability to incorporate the appropriate amount of detail and classification granularity to define specific behaviours at the sentence-level (Hung *et al.*, 2019). Likewise, Birnie *et al.* propose their system to improve risk identification and handling of both emerging and scheduled activities on an offshore oilrig. Here, an extra challenge is imposed by the inclusion of highly specialised vocabulary, reflecting the technical nature of the work. To create the dataset to meet their aims, *"8 annotators spent over 50 hours annotating thousands of workorder and incident descriptions in Prodigy's annotation tool"*. Regretfully, the specific number of articles annotated overall is not disclosed, however, Prodigy's AL capability is praised for lessening their efforts. This is achieved by training and evaluating a model in concert with the annotation task, using the intermittent outputs to help propose the most effective annotations which would benefit the model's future training (Birnie *et al.*, 2019). Comparably, Tagtog (Tagotog, 2020) offers pretrained models to automatically uncover entities, and also assists in corpus curation using AL techniques. Similarly, to Prodigy, the web-based annotation tool aims to use ML models to help the annotator spot inconsistencies in the annotation procedures, with corrections by the annotator feeding back into the model. Cejula *et al.* claim this feature allowed subject matter experts within the biomedical domain reduce overall curation times without having any knowledge of ML practices (Cejuela *et al.*, 2014).

Visualisation methods

Many current and novel approaches to document visualisation were considered prior to the beginning of this project. Najagawa, Matsui and Tsuchiya propose a method of visualising the coverage of test specification activities, a measure critical within software testing workflows. Document similarity metrics comparing the software specifications derived from software requirements and the test case descriptions are computed, and then graphed as a network of nodes and edges. The macro view of this analysis shows the nodes in the graph corresponding to each of the

specifications, and the edges denoting the strength of the similarities between them. Furthermore, the size of the nodes is adjusted in accordance with the number of assigned test case descriptions relating to that case. The system aims to help users comprehend the extent to which elements of the specification have been implemented and tested (Nakagawa, Matsui & Tsuchiya, 2017).

Document similarity analysis of Arabic text is presented in Hussein's work (Hussein, 2016), taking more of an ensembled approach to visualisation. The method demonstrates a basic but effective use of scatter plots, graphing networks, heatmaps, and dendrograms, to help explain the associations between documents post-analysis. A combination of Latent Semantic Analysis (LDA) and Singular Value Decomposition (SVD) is used to extract the distributional relationships between a very small sample of 30 documents, each roughly containing 2000 words per document. As in the previous work, the network-based representation of the documents show pair-wise similarity metrics through the modulation of the width and colour of the node edges. However, here nodes with weak similarities (below a certain threshold) have their edges eliminated, helping simplify the network and emphasize stronger connections. Simple multi-dimensional scatter plots are used to show the distance between documents in 2D space, with coloured points illustrating clustering outcomes. Dendrograms are shown to identify possible document clusters too, but more saliently, the hierarchical relationships between documents under consideration. Finally, a heat-map correlating document similarity with clustering outcomes joins both features within a simple mapping.

Sboev *et al.* propose a contextualised-semantic graph used to show nested topics within documents (Sboev *et al.*, 2015). Their approach reveals nested themes and relationships between subtopics, uncovering common themes withing strongly connected documents. Like prior approaches to networks, the node size, distance and thickness of connection all reflect the characterisation of the collection of documents. Here, instead of circular

nodes, a word from the text is used to express the nested themes. A busy overview of the connections between roughly 9000 documents is presented, and as a result, is difficult to read in any detail. However, examples presented with fewer documents succinctly reflect the relationships between documents and subtopics by highlighting the most relevant word.

Rinaldi utilises a semantic cloud, or word cloud to represent the summation of keywords extracted from documents (Rinaldi, 2013). They consider the treatment of a real document to demonstrate their system, specifically, the Wikipedia entry for Tigers. The webpage is summarised in a collection of words, the most frequent appearing larger than the less frequent.

Benhamin, Woon & Wong visualise preliminary document comparisons using an Extended Kohonen's self-organizing map (SOM) to order the vector representations of the documents and clarify relations (Benjamin, Woon & Wong, 2008). The documents occupy different regions of a square map, with similar documents appearing clustered together within the same area. Similarity scores are computed projected onto the Kohonen Map, to visualise the presence of trends, patterns, and emerging organisational structure.

Heimerl *et al.* demonstrate a hybrid synthesis of macro scale document visualization with scatter plot techniques. The proposed DocuCompass technique integrates several graphing components into an interactive exploration of a 2D document landscape (Heimerl *et al.*, 2016). Interactive scatter plots are examined with 'lenses', bounded subsections of the plot, which depict the datapoints cluster membership by colour. Mini heatmaps are overlaid adjacent to region marked by the lens, to offer a preview of the term distributions. Likewise, bar charts with corresponding colours are placed near the terms can be overlaid instead, to indicate the relative prominence of terms within the different clusters under scrutiny.

Requirements

System context

By creating both the client and server-side systems, the context in which the overall system operates is quite simple. The number of external entities that interact with the system at a given time is also limited to only one, so is also uncomplicated. Later iterations of the systems will call for a multi-user arrangement, with integrated conflict resolution between annotators and the capacity to adjudicate Gold Standard Corpora (GSC). GSC are considered by multiple experts independently, allowing for measures of inter-annotator agreement between samples to ensure the highest overall quality, however, here the focus is mainly the exploration of techniques that expedite the process of annotation generally (Wissler *et al.*, 2014).

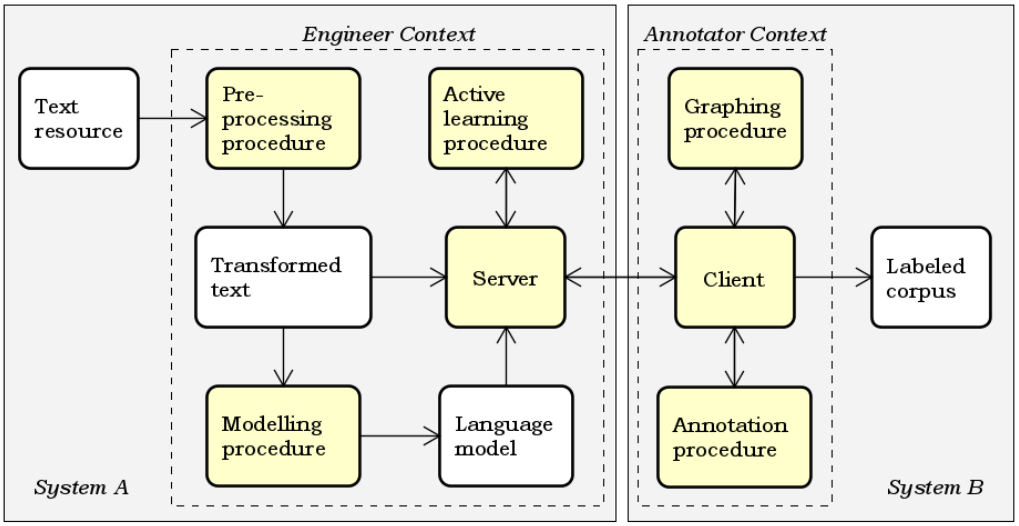


Figure 1. System context diagram

Stakeholder analysis

Given this pared-down and experimental version of the system, currently two stakeholders can be identified; but could easily be the same person. Firstly, a ML Engineer would be required to pre-process and build the ML models for use in the annotation application. In a sense, this person could be considered a maintainer or facilitator, given that the remainder of the annotation system cannot function without the text and models being in the correct form before runtime. Despite this, these processes need only be

executed and stored once, meaning that the ML Engineer is not needed so long as the text and models used for the annotation task remain correct. Secondly, the annotator is identified as an actor. This person does not need any real knowledge of the underlying models or processes to carry out their tasks, but need only to utilise their knowledge to collect and label pertinent samples.

Requirements capture

Several functional and non-functional requirements have been developed from research and an industrial partner. Typically, functional requirements describe the actions that the system must perform, while non-functional requirements describe qualitative properties with respect to how the system should operate. Within the following two tables, the MoSCoW method is used to assist in the prioritisation of these requirements. MoSCoW is an acronym of the headings Must, Should, Could and Would/Won't, and therefore enables the capture of a holistic view of the system, from the imperative to the superfluous.

Requirement	Priority	Description
F1	Must	Allow users to annotate words and sentences, using vector space models built from given text.
F2	Must	Be able to output the result of a user's annotation project in a format with is acceptable for use in ML tasks. Potentially taking the form of a custom parser/tagger that could be used within the user's workflow, or simply a formatted file.
F3	Must	Include the means to investigate, cluster, and plot text in support of annotation activities, by offering a variety of options

		for projecting and manipulating the datapoints.
F4	Must	Be a web application, to align with current trends and enhance interoperability between platforms.
F5	Should	Have some element of AL supporting the user to complete their tasks. Led mainly by the current literature, this should be achieved by either suggesting areas within the vector space that might be of interest to the user, or by recommending texts for arbitration on their classification.
F6	Should	Present annotator with analytics, describing the progress of their task.
F7	Could	Allow users to switch between or combine vector space models. For example, a model derived from user text and Stanford's GloVe Vectors could be operated on in unison.
F8	Could	Have toggleable pre-processing options. For example, including or removing punctuation and stop words.
F9	Could	Include keyboard shortcuts to speedy up binary annotation tasks
F10	Won't	Handle the processing of any digitised or scanned documents into plain text format.
F11	Won't	Be a multi-user application that requires authentication, and therefore contains

		no conflict resolution regarding discontinuity between annotators.

Table 1. Functional requirements

Requirement	Priority	Description
NF1	Must	Have a user interface which is attractive and easy to use.
NF2	Must	Feel seamless and intuitive when interacting with plots and performing annotations.
NF3	Should	Be fast and efficient in passing large amounts of data between client and server, rending the visualisations, and registering user annotations.
NF6	Should	Easy to maintain and simple to extend in its functionality.
NF7	Could	Simplify the processes of building Word2Vec models by offering automated scripts for general pre-processing and modelling activities.
NF8	Could	Be well documented, including requirements scripts to help download supplementary packages.

Table 2. Non-functional requirements

Development Methodology

Agile justification

The development of the application was supported using the Agile methodology. Firstly, the decision to follow these guidelines would lessen the impact of any changes to the requirements, should anything need to be reprioritised or refined at points throughout development. Similarly, dividing the project into manageable units allowed for a more acute focus on each of aspect of the development lifecycle per iteration. For example, taking an iterative approach can help dedicate segments of time to high-quality development, testing, feedback, and evaluation. Additionally, this approach helped with scheduling and predicting the cost in time for each feature developed. Given the pre-determined sprint lengths and concentration, this facilitated a sense of task priority early on. Finally, given that this project was partly drawn out of personal need, Agile approaches often facilitate a more critical focus on the intended user of an application in development. This can be accomplished by generating user stories as an informal means of requirements capture. The goal of a user story is to expose valuable product features in common terms, and suppress the development of superfluous components, not centred around the needs of the user. An example of user stories can be seen in later sections.

Project timeline

For reference, the following timeline was submitted during the early proposal of the project, and reflects the progression of the majority the project accurately.

Month	Objective
October 2019	• Conduct initial research into problem domain. • Develop research questions. • Submit an ethical review.

November 2019	• Submit a proposal. • Begin writing literature review. • Elicit requirements and perform requirements analysis. • Identify technology and tools required. • Begin application design. • Prototype core ideas.
December 2019	• Finalise application design. • Segment application into sprints. • Outline a testing plan. • Sprint 1.
January 2020	• Create poster. • Begin writing report. • Sprint 2.
February 2020	• Sprint 3. • Update Report.
March 2020	• Sprint 4. • Update Report.
April 2020	• Finalise Report.

Table 3. Initial project timeline.

Gantt chart

A Gantt chart is a method of visually laying out an entire project so that the big picture is easy to see, alongside task allocation, and task progress in order to hit a deadline or goal. The tasks are listed in out detail and linked together to show interdependency between tasks. Once the plan is set, the Gantt chart can be updated to reflect the progress of the task and project as whole.

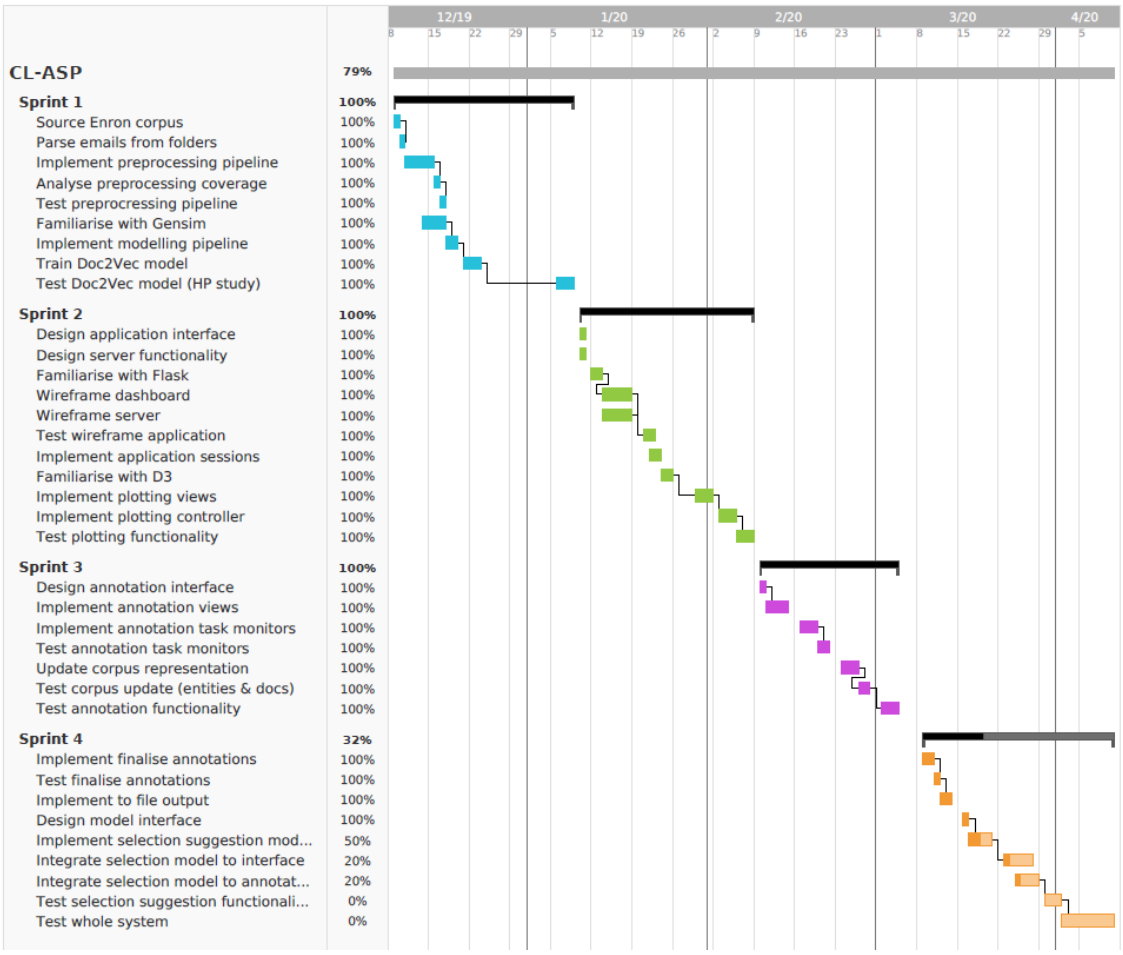


Figure 2. A Gantt chart outlining the project layout in sprints, and the current progress of the project at time of writing.

Sprint planning

Each sprint was planned to take roughly four weeks, with a leading test phase during the beginning of each development phase. In accordance with Sommerville, writing tests in this manner allowed for a firmer grasp on the intended outcomes of the system under development (Sommerville, 2016).

Alongside this, time was allocated towards the start of each subsequent sprint to allow for refactoring with an emphasis on simplicity and maintainability.

The first sprint involved implementing the processes that create the vector space model and supporting text resources. The output of this process comprises the core assets needed for the annotation application yet can be succinctly defined in a simple python script. The greatest focus and expense lay in identifying the pre-processing steps necessary to get the text into a valid format and state. Alongside this, extra time was allocated to familiarise myself with the Python module required to create the Doc2Vec model, as well as a moderate hyperparameter study.

The second sprint was focused on beginning web development. This included generating a skeletal frontend dashboard, and the supporting backend functionality. This amounted to the creation of a simple Python-based server, which could persist the models from the previous sprint, and handle basic interactivity. From here, development of the graphical components was set to begin, allowing for some extra time to get accustomed to the required libraries.

The third sprint scheduled the maturation of the graphical components, as well as the development of the manual side of the annotation processes. This sprint would also include a focus on the methods to output user annotated corpus, making testing the complete manual system possible.

Lastly, the development activities during the final sprint were centred around the implementation of the AL components, ideally merging well with the existing manual annotation system. This phase would also include the means to interact with the model on the frontend, via prompts on the user dashboard.

Design

User stores

Some aspects of the design were guided by the creation of user stories. This method was chosen due to the inherent focus on user need, brevity in capturing snapshots of the system, and applicability toward a variety of stakeholder perspectives.

Story	Description
Modelling script	I want to be able to iterate and test new versions of Doc2Vec models quickly, so make the hyperparameters immediately accessible.
Pre-processing script	I want to be able to add or remove steps from the pre-processing pipelines with ease, so modularise them.
Persist text resources and models	I want my changes to be immediately reflected server-side, within the application.
Plotting features	The plots should incorporate togglable capabilities such as showing labels, mouse tooltips, zoom, pan, etc.
Plotting customisation	The plotting capability should incorporate a variety of methods to plot and cluster the same data points.
Outputs	Outputs to file should be available in all standard formats, and conform to common labelling schemes.
Entity annotation	Entities should be able to be assigned during a document annotation task, as well as using the plots, in a straightforward way.
Active learning suggestions	Examples selected by the AL component should be presented in a way which makes them discernibly different to randomly selected

	examples. The model selection strategy could also be togglable.

Table 4. Examples of user stories, from the perspective of an engineer and an annotator.

Use case diagram

A use case diagram was generated to help design the behavioural elements of the system with more granularity. Much like the user stories, this method also allows for the users perspective to be advocated. This is accomplished by communicating the system behaviours in terms of exeternal interteractions between actors, whilst also specifying all internally visible

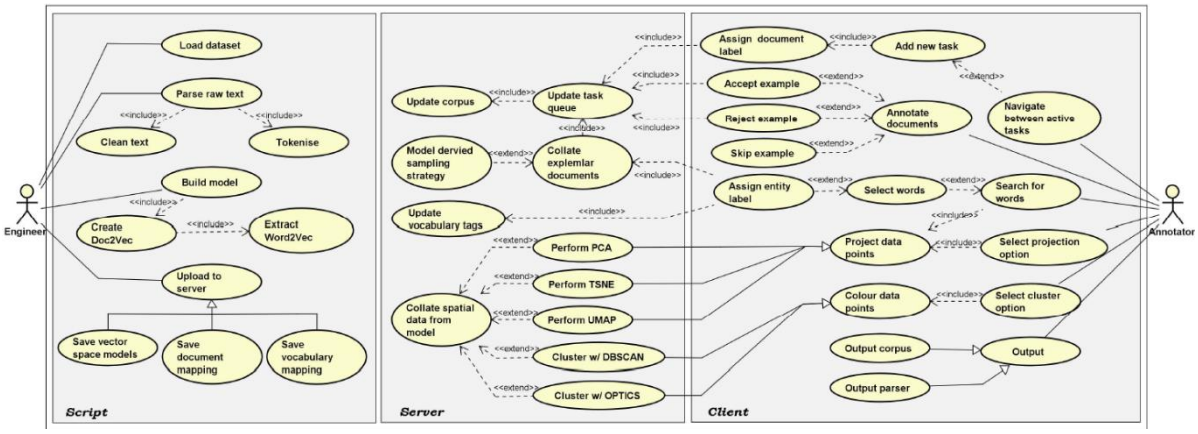


Figure 3. A use case diagram, broadly showing the behavioural design of the three main aspects of the system, and the interplay between actors.

system behavior as a result of those interactions. Actors, use cases and system boundarys are typically assigned first, afterward relationships between these elements can be addressed. The following diagram shows the complete use case design for the proposed system. This design was chosen most due to its straightforwardness of implementation. As the overall system exhibits a natual hierarchy, the dependancies between each system is clearly established, and could therefore be easy to decompartmentalise and prioritised in development.Starting from left to right, the diagram outlines the typical responsibilities of the Engineer actor. These include activities normal to NLP workflows, such as text sanitation and

tokenisation. The outcomes of the Engineer's use cases are eventually introduced to the server, for the sever to manage, and the Annotator actor to access. Although the server could possibly be designated as an actor, here it is considered as a reactionary system to the Engineer and the Annotator. Within the central system, the server's use cases revolve around querying the models, computing transforms, and maintaining the internal representations of the text. The vector space models are used to generate spatial data for use in the plotting of the word-level and document-level vector spaces. The other model is charged with generating subsets of documents for annotation based on a selection strategy, optimising for either sample diversity or uncertainty. Access and management of the structures persisting the text could arguably compel the inclusion of a dedicated database within this system, yet at this experimental level, other tools offer a similar functionality without the extra overhead in development time. Finally, the rightmost system outlines the frontend application as attended to by the Annotator. This system is responsible for rendering and displaying interactive plots, presenting the annotation tasks, and outputting any supporting information surrounding the task or the system to the Annotator.

Application interface

Owing to the investigation into NLP tools currently available, mock-up designs were generated using design cues from the applications under examination and personal preference. With an inclination toward contemporary, minimalist, and decluttered styles, these qualities inspired the design and rendering of the following figures. Figure 4 shows an application mock-up, centring around the word search and annotation views. The former is used to explore and target entities within the word space. Here, a cluster of words that exhibit structure has been discovered from a subsection of fifty words in the vector space. This specific group of words has been presented to the user due to their spatial relationship, with respect to the word provided by the user in the search bar. The cluster is encircled, and all the words are annotated with a user defined label and

optional colouring. The right side shows the chosen entities from the plot in context during an annotation session. At this step, documents containing the previously chosen words have been presented to the user, and can be annotated in two ways. Firstly, the user can manually change the

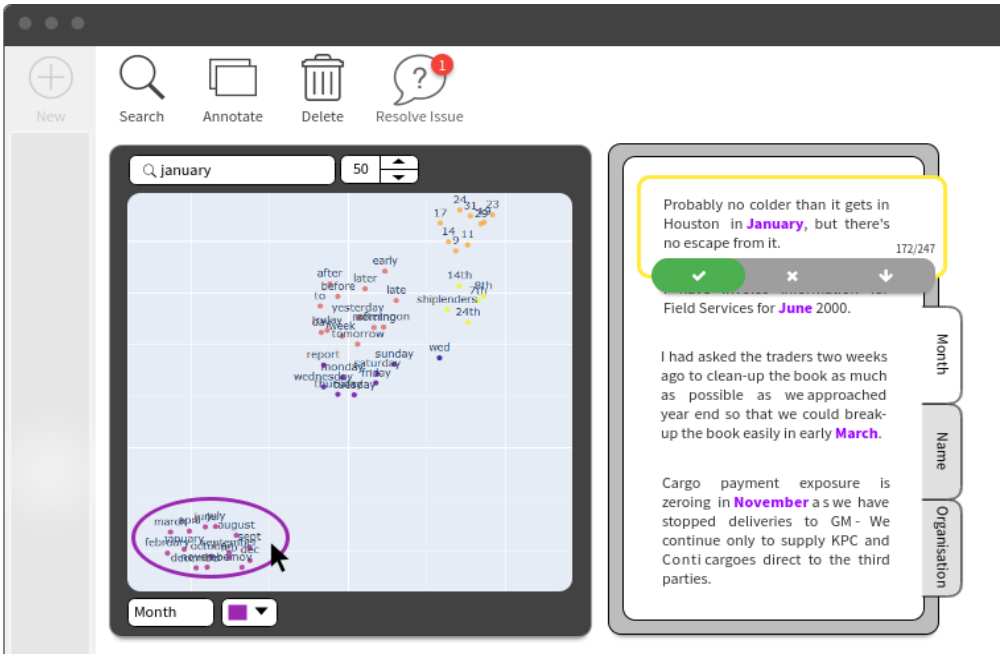


Figure 4. Mock-up design of the application, with views of the word vector search and document annotation panes, as a cluster of words is annotated.

highlighted words, or highlight other entities that appear within the text, either affirming or withdrawing a word's entity designation. Also, the user can accept, reject or skip the annotation given to a document as per its ownership to a document class, although this function zis not made clear in this rendering. Figure 5 shows another view of the application. Here, the central pane is used to display a network graph, which is centred around the sample highlighted by the user. This document-level view is derived from the similarity metrics made available by the Doc2Vec model, and reveals documents most similar to the one currently selected. The weighted edges of the graph reveal the associations between similar documents, and to what extent they are comparable by the strength of the connection. The function which generates this plot can be parameterised, allowing for the number of leaves per node, and max depth of the network produced to be adjusted.

Within the picture is another, larger graph. This broader picture of the material shows where the current document appears in relation to the remaining corpus.

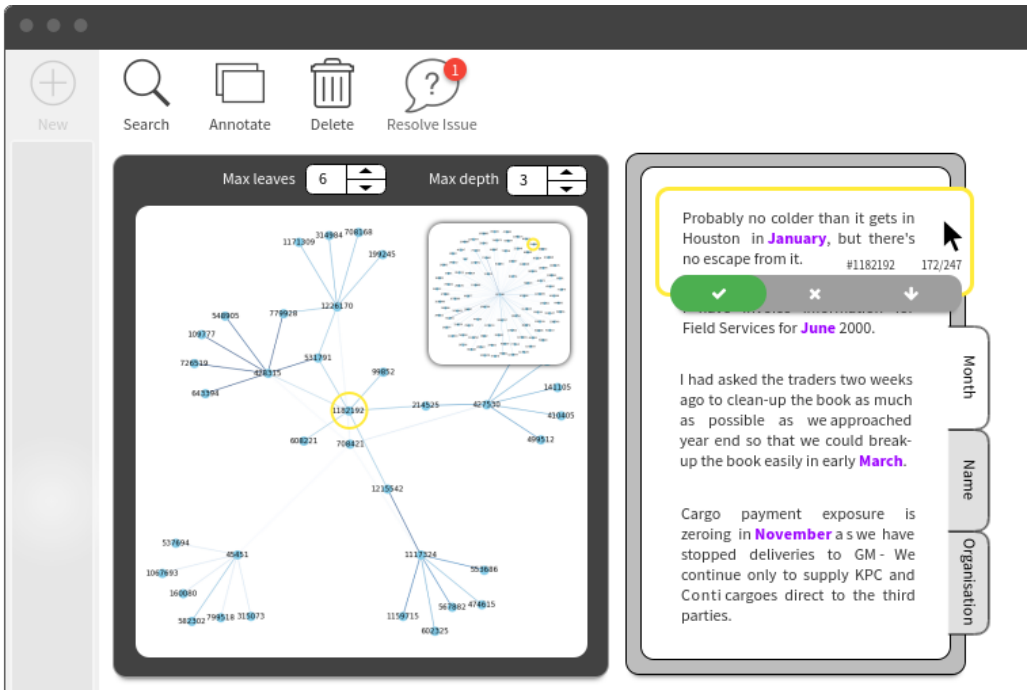


Figure 5. Mock-up design of the application, with views of the interaction between the document network graph and document annotation panes on mouseover.

Dataset selection

During the formative months of development, several common domain datasets were screened for their applicability towards the aims of this project. Amongst the few shortlisted for progression were the twenty newsgroups corpus, Reuters news corpus, and the Enron emails corpus. All commonly feature as benchmark datasets for document classification tasks, and have been used widely throughout the NLP research community. The criteria used to assess the applicability of each was the following: the corpus size, ease of access, language used, format of text, format of labelling schemes, and subject matter of text. Overall, the Enron email corpus was chosen as the central focus of the system design, due to its ability to meet

the criteria outlined. The dataset consists of over 600,000 emails, and has been deemed part of the public domain since 2003 by the Federal Energy Regulatory Commission (FERC) (Noever, 2019). Practically, this means that the corpus contains enough material to create viable models, and its use will not encroach on any intellectual property laws. The emails contained are all written in the English language, and are organised by custodian in zipped files. The content of each email is a mixture of free-form and structured text, as the body of the emails have human authors and so are subject to high potential for noise and incoherence, however, also contain the usual organisational fields associated with the anatomy of emails. Moreover, attachments and other artifacts are ignored in preference of the text, as they offer no significant benefit to the study. The emails come without any form of labelling scheme, which unlike the other two contending corpora, means that the documents are not categorised in anyway. Despite this, the content of the email is extremely suitable for use in this project for many reasons. Firstly, the likelihood that the emails contain extractable entities, such as, names, dates, times, organisations, etc is very high. Additionally, the fact the emails have human authors, and constitute a mass depiction of a realistic business environment, therefore makes it a viable platform for the proposed system to be built around. The advantage to choosing either of the other two corpora would have been their supply of target labels per each document. This would have allowed for the target labels to be treated as a control to the annotation tasks, something to aim towards and compare against. However, neither of the two contained enough samples, and only one of the two includes samples of realistic free-form text.

Implementation

Development technology

The modelling script was written in Python, mainly due to the availability of robust libraries which make ML and NLP workflows much simpler. For similar reasons, Python was also the language chosen to implement the

application server. This was achieved depending solely on the micro-framework, Flask. Mostly, plain JavaScript enables interactivity of the frontend features, however many components were quickly templated using the Bootstap.js toolkit. The creation of the visualisations and plots are supported by the D3.js library, and the highlighting of keyword entities by mark.js. Fundamentally, the ensemble of tools was geared towards the most even blend of powerful functionality and quick prototyping potential.

Pre-processing methods

Generally, pre-processing text involves techniques to normalise and remove noise from source material, although, the degrees to which these methods are applied often varies between applications and languages. Concerning the Enron corpus, the emails contain a mixture of free-form text, and semi-structured text. An example of the former would be the actual body of the message as authored by the writer, there is no structure imposed further than the grammar the author voluntarily adheres to. The latter, are the tokens that constitute the standard and optional fields filled out when an email is sent. Practically, these fields make disconnecting the header from the body of the email very straightforward. Similarly, each individual field can be extracted and included as possible a feature. For example, the contents of each 'To:' and 'From:' field can also be included in the representation of the corpus, and treated as features along with the text from the message body. For the purposes of this study, only the message id token and body are kept. Once the required text was unearthed, its case was uniformly lowered, trailing whitespace removed, and accented or special characters converted to ASCII equivalent. In this task, text that is uniform in case is preferred over the typical mixture of capitalised and non-capitalised instances, removing the chances of having multiple representations for a single word; sentence beginnings, endings, and nouns can be provided by the entity labelling scheme. Other potential transformations include expanding commonly contracted words, such as 'can't' into 'cannot', the elimination of all numeric values or punctuation, and the removal of the most common words, often referred to as the

collection of stop words. However, for this task, none of these are employed. Instead, a hybridised approach aims to preserve punctuation that conforms to custom regular expressions. Firstly, HTML tags and other mark-up artifacts are easy to identify and remove. Following that, punctuation within the remaining text is only kept if it contributes to the meta-structure of common forms alphanumeric tokens, such as, dates ('23/04/20'), the time ('14:00:00'), IP addresses ('192.168.0.1'), telephone numbers ('800-662-7662'), and so on ('A-123B/456C.789:D'). The expression used to match with and save these alphanumeric tokens considers certain punctuation marks that appear in series between valid characters. Some duplicate punctuation marks in sequence are condensed, for example '23//04//20' becomes '23/04/20', but others are rejected. In all other circumstances leading and/or trailing punctuation is removed. It is very likely that a regular expression to match all the desired cases is not feasible, especially given the sporadic nature of free-form text; alongside my limited knowledge of regular expressions generally. Yet, the system produced works well enough for the experimental aims of this project. Further pre-processing steps could include lemmatisation and stemming of the vocabulary. The former concerns gathering all inflected forms of a word into a single term, described as the word's lemma. This effectively reduces a number of common words into one, for example, the words 'writes', 'writing', and 'written', might be reduced to the base form 'write'. Stemming follows a similar path, however, normalises words by removing their suffixes, making the root term of the earlier example, 'writ' removing the suffixes 'es', 'ing' and 'ten'. Usually, the outcome of the stemmed or lemmatised representation depends on which of the many available algorithms available to do so. Regardless, both methods are considered out of scope for the purposes of this project. The set of regular expressions that drive the hybridised approach were developed and tested first against a manually selected subset of 200 documents that characterised the most complex cases, such as those within the figure above. Then tested again against randomly sampled cases. The 200 documents were chosen by first ascertaining irregular words present within the whole corpus vocabulary,

then collecting the documents where these fringe words were used. The vocabulary of the separated documents was searched again for abnormalities following the transforms, and analysis based on the number of cases that remained untreated or still atypical by the expression was carried out.

```
Teco Tap       45.000 / Enron ; 61.250 / HPL Gas Daily  LS HPL LSK IC       15.000 / Enron     ----------------------
Forwarded by Melissa Jones/Texas Utilities on  09/20/2000 10:49 AM ---------------------------   Melissa Jones 09/20/
2000 10:27 AM  To:   Charlie Stone/Texas Utilities@TU, Gary Green/Texas Utilities@TU,       daren.j.farmer@enron.com,
gary.a.hanks@enron.com,       carlos.j.rodriguez@enron.com, earl.tisdale@enron.com,       ami.chokshi@enron.com cc: S
ubject:  Enron / HPL Actuals for Sept. 19, 2000   Teco Tap       35.000 / Enron ; 71.250 / HPL Gas Daily  LS HPL LSK
IC       15.000 / Enron
```

Figure 6. Example of raw text from the body of an email.

```
teco tap 45.000 enron 61.250 hpl gas daily ls hpl lsk ic 15.000 enron forwarded by melissa jones/texas utilities on 0
9/20/2000 10:49 am melissa jones 09/20/2000 10:27 am to charlie stone/texas utilities@tu gary green/texas utilities@t
u daren.j.farmer@enron.com gary.a.hanks@enron.com carlos.j.rodriguez@enron.com earl.tisdale@enron.com ami.chokshi@enr
on.com cc subject enron hpl actuals for sept 19 2000 teco tap 35.000 enron 71.250 hpl gas daily ls hpl lsk ic 15.000
enron
```

Figure 7. Example of the same text, as a result of the using hybridised pre-processing approach.

```
['teco', 'tap', '45.000', 'enron', '61.250', 'hpl', 'gas', 'daily', 'ls', 'hpl', 'lsk', 'ic', '15.000', 'enron', 'for
warded', 'by', 'melissa', 'jones/texas', 'utilities', 'on', '09/20/2000', '10:49', 'am', 'melissa', 'jones', '09/20/2
000', '10:27', 'am', 'to', 'charlie', 'stone/texas', 'utilities@tu', 'gary', 'green/texas', 'utilities@tu', 'daren.j.
farmer@enron.com', 'gary.a.hanks@enron.com', 'carlos.j.rodriguez@enron.com', 'earl.tisdale@enron.com', 'ami.chokshi@e
nron.com', 'cc', 'subject', 'enron', 'hpl', 'actuals', 'for', 'sept', '19', '2000', 'teco', 'tap', '35.000', 'enron',
'71.250', 'hpl', 'gas', 'daily', 'ls', 'hpl', 'lsk', 'ic', '15.000', 'enron']
```

Figure 8. Example of pre-processed text as tokens, for further clarity where the token boundaries fall. Notice that dates, times, emails, and other numerical values are captured. However, as a by-product, notice the employee surname and beginning of the company 'Texas Utilities' are

Modelling methods

The modelling techniques are employed to generate a numerical representation of the input text, such that the relationships between words within the text are captured in high dimensional space, for example, synonyms, antonyms, analogies, etc. The unsupervised algorithms implemented within the proposed system, Word2Vec (Mikolov *et al.*, 2013) and Doc2Vec (Le & Mikolov, 2014) can produce or "vectorise" distributed representations of the Enron corpus using differing approaches. Approaches to Word2Vec include the Continuous Bag-of-words (CBOW), and the Skip-Gram models. Within the CBOW model, the words surrounding a target word – referred to as the context - is given as input,

then the model attempts to predict which word corresponds to the context. The Skip-Gram approach is roughly the opposite, as the model attempts to predict the surrounding context given a single word as input. The results of either is essentially a lookup table of vectors pertaining to each word in the vocabulary of corpus under analysis. Although crucial in producing the plots in the proposed system, its most notable limitation lies in its inability to address the issue of polysemy, the coexistence of multiple meanings for a single word or phrase. Recently, specialised sequence models, such as ELMo (Peters *et al.*, 2018) and BERT (Devlin *et al.*, 2018) have proven capable of handling ambiguities in NLP tasks by incorporating information about preceding/succeeding words into the decision-making processes. Doc2Vec is concerned with vectorising paragraphs or documents rather than individual words, though utilising rationally similar approaches, namely the Paragraph Vector Distributed Memory (PV-DM) and Distributed Bag-of-words (DBOW) to embed documents. PV-DM randomly samples successive words from a given paragraph, connects these words with a paragraph id, and then aims to predict a centre word. DBOW, similarly to Skip-Gram, aims to predict a probability distribution of words in a paragraph, given a randomly sampled word from the paragraph. Within the modelling script, the PD-DM method is used to produce the document embeddings. As a result of this choice, the word vectors are also computed and can be extracted from the resulting model. The usefulness of these approaches lies in their ability to group vectors of similar words and documents together in vector space. Similarities between vectors are computed using cosine similarity and form the basis of the plotting functionality. During the entity search, the word requested, along with n-number of similar vectors, are computed, and released by the model. When creating the document similarity network, the model is queried for the IDs of similar documents, along with the similarity values. As a product, the nodes and edges of the resultant graph are defined. In accordance with the initial design documentation, the maximum number of leaf nodes (n-similar) and maximum depth of this graph can be specified. This means that the model is queried for similar documents at every node when gathering

information to create the network. To prevent redundant nodes from being generated, a thresholding value can be specified. This threshold is used to prevent nodes that are too dissimilar from the root node making it into the network, additionally accelerating the compute time. This process is also maintained by the model, computing the cosine similarity between the leaf and the root node, and rejecting those that produce a similarity value smaller than the threshold. For completeness, a zero-valued threshold can be specified to produce an unconstrained network graph. For reference, during the Doc2Vec training procedure, 100-dimensional vectors are produced to reduce memory footprint of the embeddings. No minimum count of word frequency is considered when building the vocabularies representation, meaning no words are discarded even if they only appear once. This is so that for any word searched within the application, there will always be a representation for that word. The remainder of the

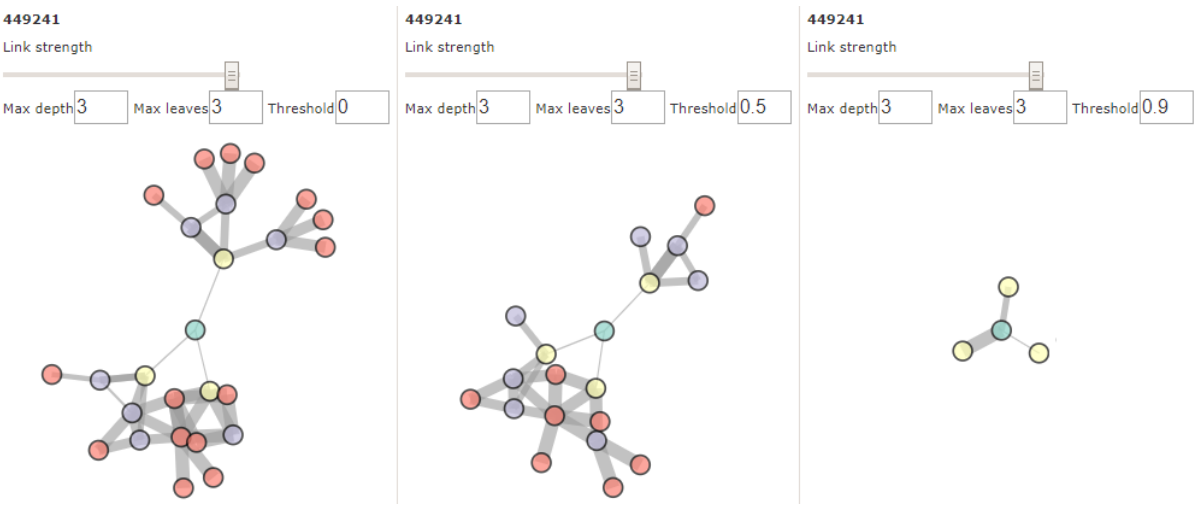


Figure 9. Example of the effect of adjusting the threshold value on graphs created from the same root document. The threshold increases left to right, from 0 to 0.9.

hyperparameters used to train the model currently supporting the application can be viewed via the modelling script. A significant parameter study has not taking place to select these values, more than judging the resulting models subjectively by eye.

Projection methods

Methods to transform and reduce the dimensionality of vectors output from the model are presented to the user. These procedures ultimately facilitate the system's ability to plot in 2-dimensions, helping support user investigations at the word-level. The techniques map each word vector onto a 2-dimensional projection, preserving the spatial qualities and therefore measures of similarity between samples from the high-dimensional (100-dimesional) space. These techniques are namely, Principle Component Analysis (PCA), T-distributed Stochastic Neighbour Embedding (t-SNE) (van der Maaten & Hinton, 2008), and Uniform manifold approximation and projection (UMAP) (McInnes *et al.*, 2018). These approaches featured a previous work, the ActiVAte system (Legg, Smith & Downing, 2019), and were considered crucial aids in facilitating the user's ability to reason about the potential relationships between samples. The same idea holds true in CL-ASP, however, are used for different ends. In this system, during entity search, the n-similar vectors returned from the model are then transformed by the chosen technique, and are then combined with the clustering methods described in the proceeding section. Both implementations of t-SNE and UMAP are non-deterministic, so have the propensity to produce slightly different plots given the same data. Furthermore, these methods can be used to cluster points in the general distribution, giving them a slight advantage over PCA; offset by the added computational cost. An example of how these approaches produce different plots given the same parameters can be seen in Table 4. Here, the vector representation of the word 'Monday' (denoted by the yellow datapoint) along with 50 of most similar vectors have been projected into 2-dimensional space using each of the three techniques available. These plots also demonstrate the clustering effects UMAP and t-SNE have on the distribution, with a clear separation between sections or regions present within both plots in comparison to PCA.

Method	Projection

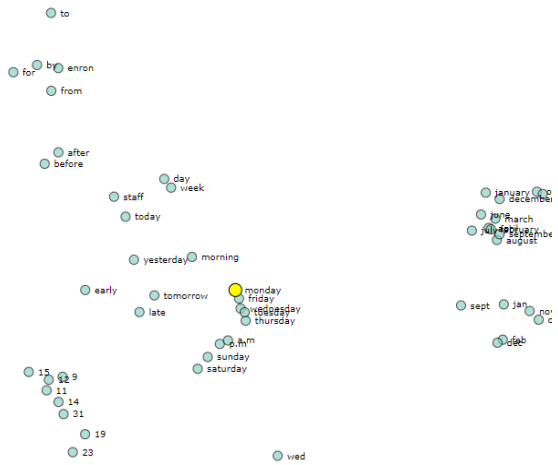
PCA	
UMAP	
t-SNE	

Table 5. Outcomes of applying each of the three available projection techniques to the same scenario (described in text).

Clustering methods

Clustering options are offered to the user during the word search processes. OPTICS (Ordering Points to Identify the Clustering Structure) (Ankerst *et al.*, 1999) and DBSCAN (Density-Based Spatial Clustering of Applications with Noise) (Ram *et al.*, 2010) can both be applied to the word vectors returned from search. The application of these techniques aims to convey or amplify some of the underlying density-based trends within the spatial data. The notion behind this is that access to such tools might aid the user locate entity boundaries, especially when investigating larger subsets of data points. These methods were chosen as neither require the user to

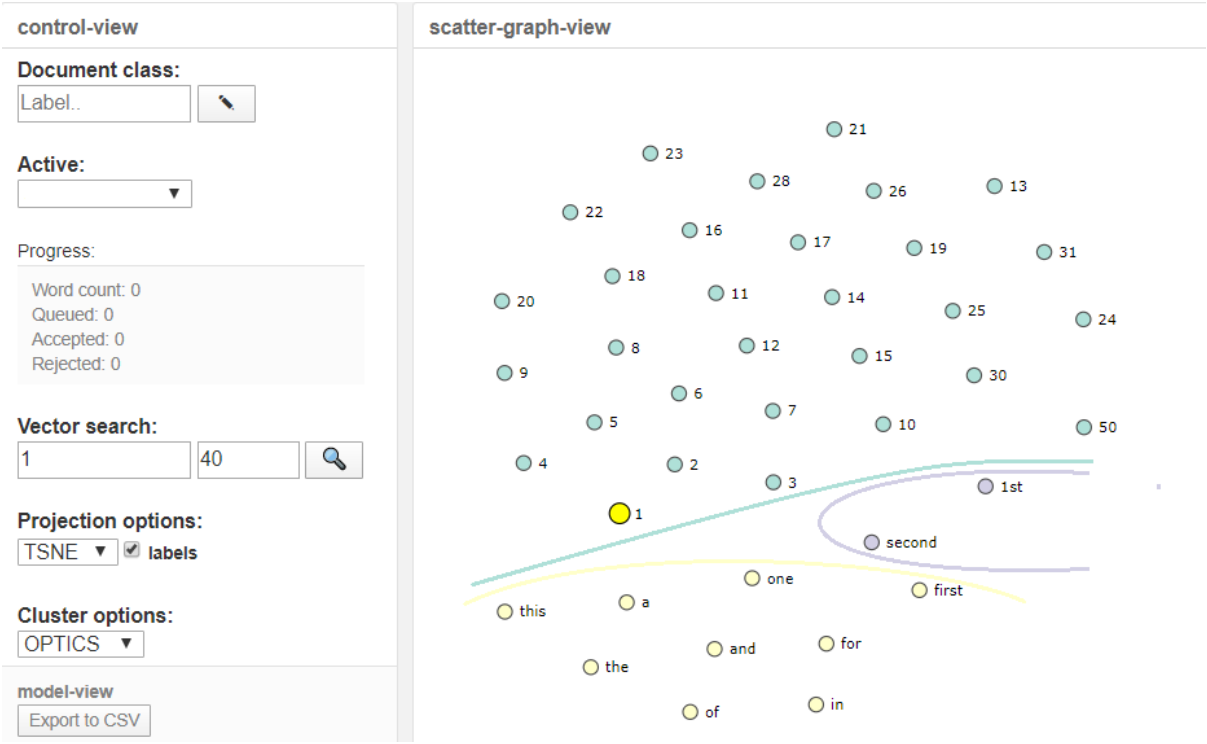


Figure 10. Example of an entity search procedure. The vector representation for the word '1' is returned, along with 40 words within its vicinity. The resultant vectors are then transformed using the t-SNE and OPTICS methods.

specify the number of clusters apparent in the data a priori, opposed to similar approaches like k-means clustering. Similarly, both approaches can find arbitrarily shaped clusters. Some parameterisation is required for these approaches to work optimally, and therefore is not an ideal solution. However, further work could allow the user to manipulate these values, or

even have them automatically calibrate to the plot. The same scenario that was used to produce the plots shown in Table 4 is re-examined in Table 5, however, utilises the clustering techniques to colour the graphs generated. In comparison, clusters within the plots shown in Table 5 are further intensified and much easier to spot with the addition of the colouring. This is particularly true for the t-SNE plot, which can often look flat when used to transform a small number of samples.

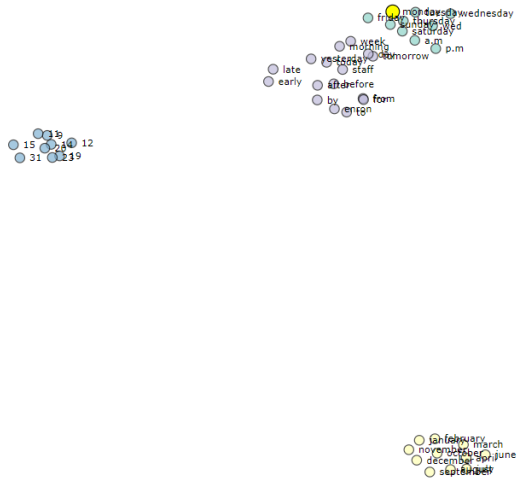
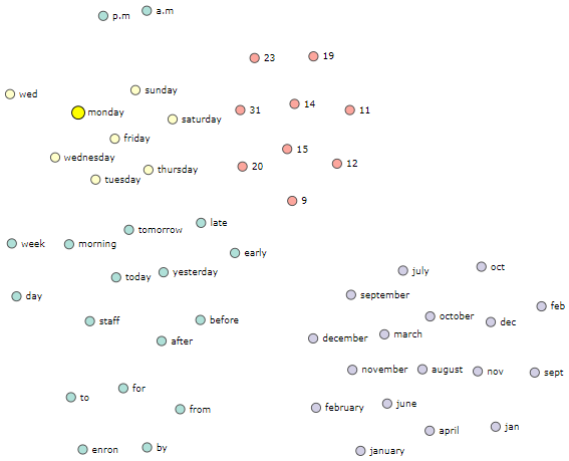
Method	Projection
UMAP & DBSCAN	
t-SNE & OPTICS	

Table 6. Outcomes of using a each of the two available clustering methods, alongside projection techniques.

Results

Issues stemming from the incompleteness of the development of the tool make comparing it against other tools and methods largely impossible at this stage. However, the main functionality of the whole system can be examined in the following figures. In this representative scenario, the document-level task calls for the identification and collection of documents whose contents involve text that describes a meeting. The active document-level class is set to 'meeting', and provides the motivation for the secondary

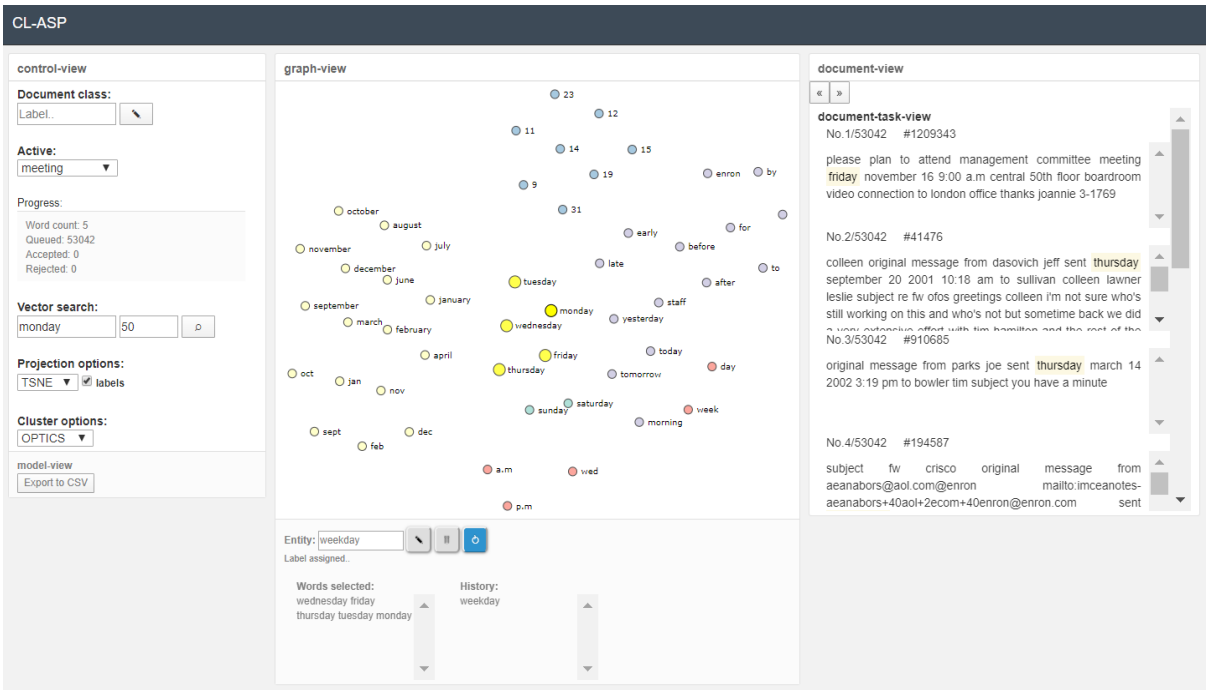


Figure 11. Whole system view. From left to right, the system is derived of three panels. The control view hosts the controls and progression metrics for the document-level annotation task, and the controls for the word search tasks. The central pane, or graph view, is used to display and interact with plots. The bottom portion is used to log the user derived relationships between words and assigned entity tags. The document view panel shows the documents sampled as a result of word search. This panel dynamically extends into another, to include documents added by way of interacting with nodes in the network plots.

task of collating a set of words which most likely belong to documents within that class. It is fair to reason that most corporate meetings are likely to happen on a weekday, therefore, the word 'Monday' is searched within the system. The corresponding word vector, along with 20 words within its vicinity are returned. As a product of the projection and cluster options, the

plot in Figure 11 shows some well separated clusters of words, coloured as if members of separate groups. The words 'Tuesday', 'Wednesday', 'Thursday' and 'Friday', are selected, and accompany the previously selected word, 'Monday'. To accelerate the selection process, the lasso tool is used to select points by allowing the user to draw a free-form capture area. This

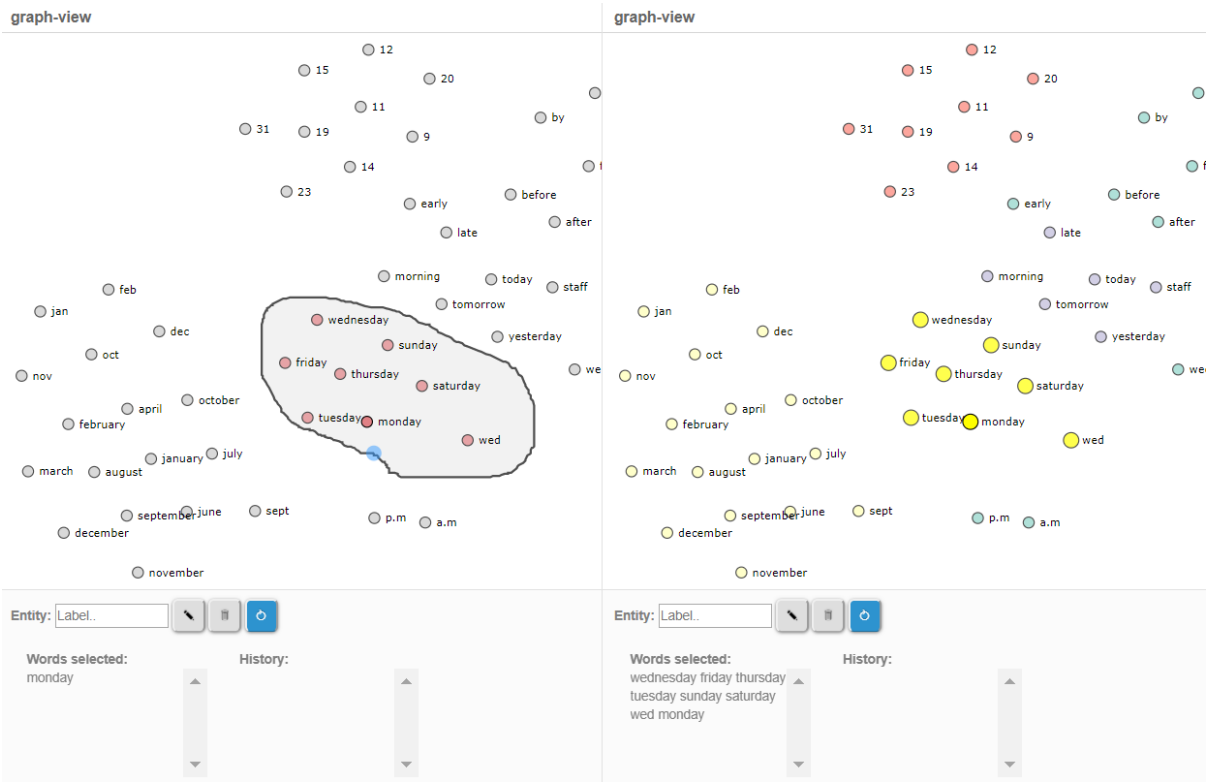


Figure 12. The lasso functionality allows the user capture and select multiple words simultaneously. Once selected, the words within the captured group are emphasized with bright colours and an increase in size. Also, the words are registered in the box below the plot for reference.

functionality is demonstrated in Figure 12. From here, the entity label 'weekday' is assigned to this group of words. Once this is applied, the system presents a collection of randomly sampled documents containing these words in the document task view. As can be seen in the right panel of Figure 11, the relevant words are highlighted within the text to help quickly judge the appropriateness of their use within the context of a meeting. As the users accepts, rejects, or skips examples based on these criteria, the users progress is shows in the progress section to the left of the screen. Demonstrated in Figure 13, the user can also choose to generate a graph using a given document with ranging levels of granularity and control.

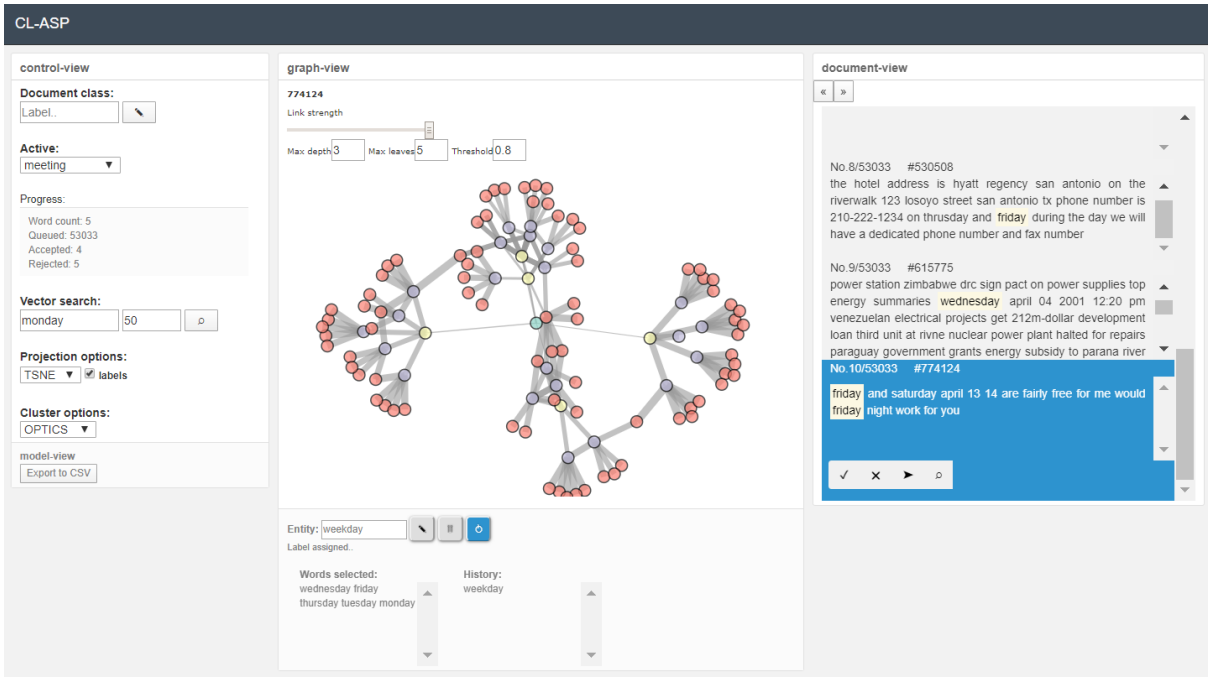


Figure 13. A document from the document view panel is chosen to become the source of a network graph. The sampled document lies at the root, with leaf nodes appearing from it. This process continues for all leave nodes until the specified depth is reached. The number of nodes within the graph can be constrained given their relative similarity to the root node, effectively suppressing the dissimilar documents from being included in the plot. The root node appears in green, nodes at the next level are yellow, and the terminal nodes are red. Nodes from each level are coloured differently to express this relationship.

Nodes within the graph can be presented as documents by interacting with the nodes. Once selected, corresponding documents are displayed in the

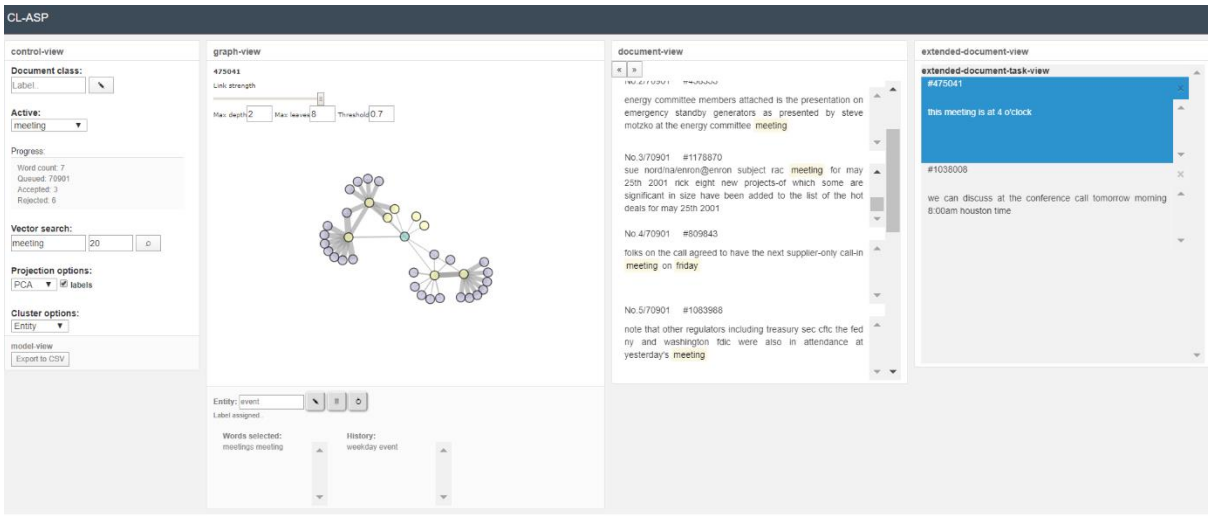


Figure 14. A view of the extended task panel. The collection of words used to generate candidate documents is added to. The history section of the middle panel tracks the entities currently contributing to the search. After a network was plotted using a sample from the initial pool, a leave node has been selected, and the corresponding document appears in the right-most panel. This document is then used to generate another network plot.

extended document view, allowing them to be examined, annotated and graphed as usual in an extended panel. As shown in Figure 14, the words 'meeting' and 'meetings' are added to the criteria and are given the entity tag 'event'. This updates the documents presented to the user, and many will now include a combination of words under tags 'weekday' and 'event'. Representations like this can be built up indefinitely, with the ability to add and remove words contributing to document search and entity tagging. At the culmination of the task, once the user is satisfied, they can discharge a dataset based on their work, as demonstrated in Figure 15. Each row is comprised of the following:

1. A reference (the index) to the original dataset.
2. A reference of where the proceeding sentence features in the original text.
3. The pre-processed text comprising the sentence/document.
4. The IOBs tags at positions corresponding to the words in the previous sentence.
5. The class label given the sentence/document.

doc_id	sen_id	text	iob	label
22796	7	jeff will be holding an info	O, O, O, O, O, O, O, O, B-WEEKDAY, O, O, O, O, O, O, O, O	MEETING
38501	0	this meeting is at 4 o'clock	O, B-EVENT, O, O, O, O	MEETING
19515	6	this friday october 26th 11	O, B-WEEKDAY, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O	MEETING
3366	12	at that point i expected th	O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O,	MEETING
84988	9	the purpose of our meetir	O, O, O, O, B-EVENT, O, O, O	MEETING
99403	0	monday's ooc staff meetir	O, O, O, B-EVENT, O, O, O, O, O, O, O, O, O, O, O, O, O, O	MEETING
65132	21	folks on the call agreed to	O, O, O, O, O, O, O, O, O, O, O, B-EVENT, O, B-WEEKDAY	MEETING
74695	45	the meetings are part of a	O, B-EVENT, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, (	MEETING
50425	0	to all enron employees ple	O, O, O, O, O, O, O, O, O, O, B-EVENT, O, O, O, O, O, B-WEEKDAY, O, C	MEETING
17881	65	research open season req	O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, O, B-EVENT, O, O, (	MEETING

Figure 15. CSV file output as a result of the annotation task.

Evaluation

With respect to the functional requirements outlined in table 1, the top priority requirements F1, F2, F3, F4 were absolutely met. Sadly, the arguably more interesting parts of the system remain incomplete, specifically, the integration of the planned AL components. It is with great frustration that I have not been able to present the full scope of my vision,

and push this project into more exciting territory. This shortcoming is largely down to a failure of planning. Specifically, not mitigating the effects of overrunning sprints, not allocating enough time for the project's evaluation against competing tools, nor the beginning of this writeup. Secondly, the failure to implement the remainder of the system is also factor of circumstance. At the time of writing, the inevitable disruption that the COVID-19 pandemic has wrought upon the globe, has also negatively impacted the completion of the latter part of this project. As for the Non-functional requirements defined in table 2, my assessment is that all these aims were met, even within the current skeletal system.

Limitations

The system has many limitations to overcome. Several items concerning the systems main functionality remain undeveloped, lessening the projects purpose and validity in the process. Of great importance is the glaring fact that many views of the data remain underdeveloped. For example, as seen in Figure 5, overlaid network graphs describing document similarity in greater context were designed. Regretfully, this and many other ideas were developed in isolation, but remain unimplemented in the system as delivered. Additionally, the system has no method of saving an annotation session. Given the potential length of these tasks, I see this as quite a major limitation. Equally, the fact that it is single user system is quite problematic, especially as this denies the cultivation of a GSC. Without this, the system fails to breach the issue of inter-annotator inconsistencies, bias, and resolution. This capability would need to be established before this system could be used in any professional context, but as previously stated, scaling the system was considered out of scope. My perspective is that the system proposed shows promise, and is still uniquely positioned amongst the current annotation tools on offer. This is largely down to the systems approach to user interaction. Being able to search for terms, and explore the latent space with granularity is a feature non-existent in other annotation tools. This is potentially due to the perception that these processes seem overly technical, or even counterproductive given the

potential to confront distracting or noisy samples within an investigation into the feature spaces. Yet, this factor is largely down to the quality of the text used, with respect to how the features were engineered or amplified during the pre-processing stages, and also the quality of the resultant model in its ability to accurately produces similar words and documents. The development process could have been improved by seeking out more feedback about the system from external sources. It was made plain to me too late in development how underutilised my supervisor had been during this process; perhaps that is obvious given the quality of the product.

Conclusion

Despite not be able to deliver the full system, it is my belief that the proposal of integrated interactive data views and AL approaches to annotation remains a valid and exciting avenue of research. Notwithstanding the shortcomings of the system, I am mostly happy with the product so far, and will continue to develop it into what I had envisaged. I feel that the process could have been vastly improved by seeking out more feedback about the system from external sources. Two consultation studies were planned to gain both an industrial and academic's perspective on the system, however, inhibitions stemming from its underdevelopment prevented me from take advantage of their time. Likewise, it was made plain to me late in development how underutilised my supervisor has been during this process, and perhaps that is obvious given the completeness of the product. Future work would revolve around objectively evaluating the usefulness of a wide range of integrated visualisation techniques, more than what has been presented within this system. Further than this, optimised methods of sample selection using AL should be assimilated and evaluated in its ability to present more salient documents to the user during annotation tasks.

References

Ankerst, M., Breunig, M., Kriegel, H. & Sander, J. (1999) OPTICS: ordering points to identify the clustering structure [online]. *Proceedings of the 1999 ACM SIGMOD international conference on Management of data - SIGMOD '99*. [Accessed 23 April 2020].

Benjamin, C., Woon, W. & Wong, K. (2008) A graphical and convenient tool for document comparison and visualization [online]. *2008 International Conference on Computer and Communication Engineering*. [Accessed 23 April 2020].

Birnie, C., Sampson, J., Sjaastad, E., Johansen, B., Obrestad, L., Larsen, R. & Khamassi, A. (2019) Improving the Quality and Efficiency of Operational Planning and Risk Management with ML and NLP [online]. *SPE Offshore Europe Conference and Exhibition*. [Accessed 23 April 2020].

Cejuela, J., McQuilton, P., Ponting, L., Marygold, S., Stefancsik, R., Millburn, G. & Rost, B. (2014) tagtog: interactive and text-mining-assisted annotation of gene mentions in PLOS full-text articles [online]. *Database*. 2014. [Accessed 23 April 2020].

Dai, H., Lai, P., Chang, Y. & Tsai, R. (2015) Enhancing of chemical compound and drug name recognition using representative tag scheme and fine-grained tokenization [online]. *Journal of Cheminformatics*. 7 (S1), . [Accessed 23 April 2020].

Devlin, J., Chang, M., Lee, K. & Toutanova, K. (2018) *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* [online]. [Accessed 23 April 2020].

Heimerl, F., John, M., Han, Q., Koch, S. & Ertl, T. (2016) DocuCompass: Effective exploration of document landscapes [online]. *2016 IEEE*

Conference on Visual Analytics Science and Technology (VAST). [Accessed 23 April 2020].

Hung, B., Muramudalige, S., Jayasumana, A., Klausen, J., Libretti, R., Moloney, E. & Renugopalakrishnan, P. (2019) Recognizing Radicalization Indicators in Text Documents Using Human-in-the-Loop Information Extraction and NLP Techniques [online]. *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*. [Accessed 23 April 2020].

Hussein, A. (2016) Visualizing document similarity using n-grams and latent semantic analysis [online]. *2016 SAI Computing Conference (SAI)*. [Accessed 23 April 2020].

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K. & Dyer, C. (2016) Neural Architectures for Named Entity Recognition [online]. *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. [Accessed 23 April 2020].

Le, Q. & Mikolov, T. (2014) Distributed Representations of Sentences and Documents [online]. *ICML'14: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 June 2014 Pages II-1188–II-1196*. [Accessed 23 April 2020].

Legg, P., Smith, J. & Downing, A. (2019) Visual analytics for collaborative human-machine confidence in human-centric active learning tasks. *Human-centric Computing and Information Sciences*. 9 (1), . [Accessed 23 April 2020].

McInnes, L., Healy, J., Saul, N. & Großberger, L. (2018) UMAP: Uniform Manifold Approximation and Projection [online]. *Journal of Open Source Software*. 3 (29), pp. 861. [Accessed 23 April 2020].

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013) Efficient Estimation of Word Representations in Vector Space [online]. *CoRR abs/1301.3781*. [Accessed 23 April 2020].

Nakagawa, H., Matsui, S. & Tsuchiya, T. (2017) A Visualization of Specification Coverage Based on Document Similarity [online]. *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. [Accessed 23 April 2020].

Neves, M. & Ševa, J. (2019) An extensive review of tools for manual annotation of documents [online]. *Briefings in Bioinformatics*. [Accessed 23 April 2020].

Noever, D. (2019) *The Enron Corpus: Where the Email Bodies are Buried?* [online]. [Accessed 23 April 2020].

Pennington, J., Socher, R. & Manning, C. (2014) GloVe: Global Vectors forWord Representation [online]. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), October 25-29, 2014, Doha, Qatar. Association for Computational Linguistics*. pp. pages 1532–1543. [Accessed 23 April 2020].

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K. & Zettlemoyer, L. (2018) Deep Contextualized Word Representations [online]. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. [Accessed 23 April 2020].

Prodigy (2020) *Prodigy · An annotation tool for AI, Machine Learning & NLP*. Available from: https://prodi.gy/ [Accessed 23 April 2020].

Ram, A., Jalal, S., Jalal, A. & Kumar, M. (2010) A Density Based Algorithm for Discovering Density Varied Clusters in Large Spatial Databases [online].

International Journal of Computer Applications. 3 (6), pp. 1-4. [Accessed 23 April 2020].

Ramshaw, L. & Marcus, M. (1999) Text Chunking Using Transformation-Based Learning [online]. *Text, Speech and Language Technology*. pp. 157-176. [Accessed 23 April 2020].

Ratinov, L. & Roth, D. (2009) Design challenges and misconceptions in named entity recognition [online]. *Proceedings of the Thirteenth Conference on Computational Natural Language Learning - CoNLL '09*. [Accessed 23 April 2020].

Rinaldi, A. (2013) Document Summarization Using Semantic Clouds [online]. *2013 IEEE Seventh International Conference on Semantic Computing*. [Accessed 23 April 2020].

Sboev, A., Moloshnikov, I., Gudovskikh, D. & Rybka, R. (2015) Visualization of Subtopics of the Thematic Document Collection Using the Context-Semantic Graph [online]. *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*. [Accessed 23 April 2020].

Sommerville, I. (2016) *Software Engineering, Global Edition*. NOIDA: Pearson Education Limited.

spaCy (2020) *spaCy: Industrial-Strength Natural Language Processing* Available from: https://spacy.io [Accessed 23 April 2020].

Tagtog (2020) *Tagtog · AI-enabled Text Annotation Tool | PDF, Markdown, CSV, html, tweets, & many more Document types*. Available from: https://www.tagtog.net/ [Accessed 23 April 2020].

van der Maaten, L. & Hinton, G. (2008) Visualizing High-Dimensional Data Using t-SNE [online]. *Journal of Machine Learning Research*. 9pp. 2579-2605. [Accessed 23 April 2020].

Weischedel,, R., Palmer, M., Marcus, M., Hovy, E., Pradhan, S., Ramshaw, L., Xue, N., Taylor, A., Kaufman, J., Franchini, M., El-Bachouti, M., Belvin, R. & Houston, A. (2013) *OntoNotes Release 5.0*. 5th edition. Available from: https://catalog.ldc.upenn.edu/LDC2013T19 [Accessed 23 April 2020].

Wissler, L., Almashraee, M., Monett, D. & Paschke, A. (2014) The Gold Standard in Corpus Annotation [online]. *Conference: 5th IEEE Germany Student Conference At: Passau, Germany*. [Accessed 23 April 2020].

Yang, J., Liang, S. & Zhang, Y. (2018) Design Challenges and Misconceptions in Neural Sequence Labeling [online]. *Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, New Mexico, USA, August 20-26, 2018.*. pp. pages 3879–3889. [Accessed 23 April 2020].

Table of figures

Table of tables