Using information extraction and machine learning to determine "Further Work" in Artificial Intelligence research papers

Steven Smith – P13167208

IMAT5314 Dissertation

Overview

- Introduction
- Literature Review
- Data Construction
- Model Construction
- Analyses
- Conclusions
- Further Work
- References

Introduction

- For new research, researchers review a large number of papers
- Keshav (2007) – “How to read a paper” – First pass
  - Read abstract and introduction
  - Skim read or flick through middle pages
  - Read conclusion
  - In-depth reading of paper by beginning researcher – five or six hours
- Search engines – arXiv.org and scholar.google.co.uk
  - meta-data (title, authors, references, etc.) & abstracts
  - No details on extending or furthering work
    - May give direction or spark new ideas

Introduction – Current Research (Brief)

- Research Paper Information Extraction
  - Content and meta-data extraction - TKACZYK et al. (2015).
  - Figure and table extraction - Clark and Divvala (2016).
  - Key Phrase Extraction & Text Summarisation - Popova and Danilova (2014), (Lopez and Romary, 2010), Lloret, Romá-Ferri and Palomar (2013).
  - Knowledge Extraction - Jain et al. (2015), (Song et al., 2015).
- This is an active area of interest.

Introduction – The aims of this research

- Question 1 - Can further work be identified and retrieved from research papers using machine learning?
- Question 2 - Which is the best classifier to classify further work sentences?
  - Multinomial Naïve Bayes (MNB)
  - Support Vector Machines (SVM)
  - Convolutional Neural Network (CNN)
- Question 3 - Which features will train the best model?

Introduction – The approach

- Gather some research papers in digital format
- Extract the text and section headings from the papers
- Clean the extracted text
- Manually annotate the text as “further work” or “not further work”
- Extract features from the annotated text
- Train candidate models with these features
- Select the best model

Introduction – The scope

- Research Papers from arXiv.org
  - “Physics, Mathematics, Computer Science, Quantitative Biology, Quantitative Finance and Statistics”
- Only Computer Science – Artificial Intelligence Papers (over 11,000)
- Only papers with an identifiable conclusion/further work sections
- Only papers with fifteen or less conclusion/further work sentences

Introduction – Experimental Environment

- Machine
  - Intel Core i7-6800K 3.40GHz (Broadwell-E/EP)
  - 64GB DDR4 RAM
  - 2 x NVIDIA GeForce GTX 1080 8GB
  - Samsung SSD 850 EVO 1TB, Samsung SSD 960 EVO 1TB, Western Digital Black 4TB
  - Ubuntu Linux 17.04
- Development Environment
  - Python 3.5, Jetbrains PyCharm, Anaconda, scikit-learn, NumPy, SciPy, pandas, lxml, Requests, gensim, Natural Language Toolkit, Beautiful Soup, TensorFlow, imbalanced-learn
  - Notepad++, CERMINE, Stanford NLP POS Tagger

Literature Review – Information Extraction

- Meta-data and Reference/Bibliography Extraction
  - CERMINE – “Content ExtRactor and MINEr” – Tkaczyk et al. (2015)
    - Extracts text, headings, titles, authors, references and figures.
    - Zone classification using an SVM, Rule-based meta-data extraction, Reference extraction using K-means clustering and conditional random fields.
    - Tables are not handled separately, end up embedded in the text.
- Figure/Table extraction
  - PDFFigures 2.0 – Clark and Divvala (2016)
    - Identify captions using list of starting key phrases
    - Region identification - body text, figure text, captions, graphics
    - Novel features - Statistics; font size, line widths, overlaid graphics, headings, margins

Literature Review – Information Extraction

- Key Phrase Extraction
  - Popova and Danilova (2014)
    - Identify key phrases in the entire paper using abstracts to reduce scope
    - Abstract only limits issues of ranking the best key phrases as there are less to choose from.
    - Reduce scope by limiting phrase structure, e.g. verb, noun or adjective, verb, noun
    - Improved the state of the art to F1-Scores of 0.31, 0.27 and 0.23 on the test datasets.
- Text Summarisation
  - Lloret, Romá-Ferri and Palomar (2013)
    - Two part engine – Text summarization and information compression
    - 38%, 44% and 32% of the time assessors agreed or strongly agreed with the three different test summaries generated
    - Still not possible to replace human summarizer

Literature Review – Information Extraction

- Knowledge Extraction
  - Extracting knowledge from Genomic research - Jain et al. (2015)
    - Phenotype, ethnic group and the stage of the study (initial or replication)
    - Precision = 0.8746 for Phenotype extraction (No recall or F1-Score)
    - Precision = 0.7893, Recall = 0.8757, F1-Score = 0.8302 for ethnic group and stage
    - Not clear if this is good enough to replace human intervention, but would offer support for humans
  - Entity and relation text mining – PKDE4J – Song et al. (2015)
    - Entity and relation extraction for public knowledge discovery – PKDE4J
    - Biomedical entities: Cell types, components, processes/functions, diseases, drugs
    - Relation between entities: Positive, negative, neutral, plain
    - Uses multiple dictionaries and custom rules to extract entities and relations
    - Dictionary Based Extraction performs poorly so they use lemmatization, normalization and similarity measures to do approximate string matching.
    - Pipeline - Abbreviation resolution, sentence splitting, POS tagging, lemmatization, normalization, entity extraction and relation extraction.
  - Both these papers use a committee of weak classifiers to strongly classify their data.

Literature Review – NLP Feature Selection

- Stop word removal
- Bag of words – vector of word counts
  - Application - Wang & Manning, 2013; Popova and Danilova, 2014; Zhang and Wallace, 2015; Jain et al.,2015
  - Counts, TF-IDF, Unigrams, Bigrams, Trigrams
- Word2Vec - Mikolov et al. (2013)
  - Represents a word using the words before and after in an n-dimensional vector space.
  - Vector for King – Vector for Woman = Vector for Queen
  - Application - Kim (2014), Zhang and Wallace (2015), Le, Ceriasra and Denis (2017), Zhang, Zhao and LeCun (2015) and Liu, Qiu and Huang (2016).

Literature Review – NLP Feature Selection

- Word convolution
  - Word2Vec matrix of words in sentence - Kim (2014)
  - Further investigated by Zhang and Wallace (2015)
  - 3 parallel convolution layers
  - 1-max pooling (Boureau, Ponce and LeCun, 2010)
- Character learning - Zhang, Zhao and LeCun (2015)
  - 70 characters – 26 lowercase letters, 10 digits, 33 symbols/punctuation
  - One hot encoding – 70 dimension vector
  - 50 character sentence – 70 rows by 50 columns
  - Fixed length sentence – filled with zero vector
  - Requires large amount of data due to deep learning – 9 layers

Literature Review – NLP Feature Selection

- Custom domain/specific features
  - CERMINE uses 42 hardcoded rules to classify a token - Tkaczyk et al. (2015)
  - Rules to determine whether text belongs to body or image - Clark and Divvala (2016)
  - Dictionary of domain specific words and the distance of words from the beginning of the document - Jain et al. (2015)

Literature Review – Sentence Classifiers

- Multinomial Naïve Bayes
  - Applied in Wang and Manning (2012), Kim (2014)
- Support Vector machines – Cortes and Vapnik (1995)
  - Applied in Kim (2014), Zhang, Zhao and LeCun (2015), Zhang and Wallace (2015)
  - NBSVM – Wang and Manning (2012)
  - Final classification of features by SVM, Jain et. al (2015)
  - Sentiment of every sentence classified using an SVM to determine the sentiment of the entire document - Wang et al. (2014)

Literature Review – Sentence Classifiers

- Convolutional Neural Networks
  - Shallow & wide – Kim (2014)
  - Character learning & deep learning - Zhang, Zhao and LeCun (2015)
  - Shallow & wide vs Deep Learning comparison - Le, Cerisara and Denis (2017)
    - Found Deep Learning is not more effective
- Long Short-Memory
  - LSTM network (Hochreiter and Schmidhuber, 1997) builds on RNNs (Elman, 1990) ability to learn sequences of data.
  - Variable length sequences, multi-task learning - Liu, Qui and Huang (2016)
  - Discriminative vs generative models - Yogatama et al. (2017)
    - Generative models cope better with a changing text distribution
  - The ability to remember information from earlier in the sentence
    - More information rich than bag of words (Yogatama et al., 2017)
- Deep learning needs large datasets - Liu, Qui and Huang (2016)

Dataset Construction – PDF Acquisition

- https://arXiv.org – Scientific Research Paper archive – Cornell University Library
  - “is an e-print service in the fields of physics, mathematics, computer science, quantitative biology, quantitative finance and statistics”
- Conclusion documents and sentences xml file

```
<docs>
  <doc key="1205.3336" processed="True" num-sentences="1" id="1">
    <sentence id="1" is-fw="False">Once … previous results.</sentence>
  </doc>
  <doc key="0805.1153" processed="True" num-sentences="2" id="2">
    <sentence id="1" is-fw="False">Some … database update.</sentence>
    <sentence id="2" is-fw="True">In … future works.</sentence>
 </doc>
</docs>
```

Data Construction – Bulk download PDFs

- PDFs can be downloaded individually
  - https://arxiv.org/abs/math/0011059 - Old format
  - https://arxiv.org/abs/1707.05308 - New format
  - Actions - abs, pdf, ps, html, e-print (source file zip), other (all formats)
  - Fair usage policy – may block bulk downloading using these URLs
- "All" the PDFs in .tar files – Amazon S3 Requester Pays Bucket (~$80 for ~760GB of data) ~500MB each
  - http://s3tools.org/s3cmd - command line interface to download remote files
  - arXiv_pdf_0001_001.tar – January 2000 (0001) file number 001
  - arXiv_pdf_0001_002.tar – January 2000 (0001) file number 002
  - Can determine which month from identifier but not which file.

Data Construction – Acquire Meta-data

- Three meta-data APIs
  - Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH)
    - A bit too verbose
  - arXiv API
    - Concise specific XML format
    - http://export.arxiv.org/api/query?search_query=cat:cs.AI&start=0&max_results=10000&sortBy=submittedDate&sortOrder=ascending
  - RSS feed
- XML format meta-data giving information about research papers
  - Id, title, published, comment, journal_ref, authors, category, primary_category
- XML format saved to a pandas DataFrame
  - Easily manipulated and easily serialized to disk
  - Additional columns - root_category, key, version, filename, new_key_format, pdf_exists, xml_exists, html_exists, zones_exist, conclusion_xml, num_sentences
- Loop over entries and attempt to extract PDF from tar file
  - easy?

Data Construction – Download missing PDFs

- Some PDFs not contained in the tar files
  - Don’t know why they are not there!
- Make a request to the https://arXiv.org/pdf/... URLs
- If still can’t be found create a file with .missing.html file extension
  - Some of the Research Papers do not have a PDF version
  - To stop retrying the download

Data Construction – PDF to XML - CERMINE

- CERMINE
  - Formats
    - NLM JATS – NIH format for structuring journals
    - Raw Text
    - TrueViz (Overly rich XML format)
    - Images
    - Text zones - Flatish XML – Zone labels and text
  - Java JAR application callable from command line to extract a directory
  - Python script calling the executable
    - Determines if the executable returns an error – characters are not legal xml
    - Creates XML stub if error occurred
    - Calls CERMINE executable again until no files left to process

Data Construction – Extract conclusions

- CERMINE Text zone format
  - <document>
    <zone label=”ZONE_TYPE1”>some text</zone>
    <zone label=”ZONE_TYPE2”>some more text</zone>
    </document>
  - Labels – BODY_CONTENT, BODY_HEADING, MET_TITLE, “MET_”, GEN_OTHER, GEN_REFERENCES
    - Not perfect – sometimes text in wrong zone – Out of scope to address this
  - BODY_HEADING – Section titles to look for conclusions/further work
    - .*(conclus|work|research|direction).* - rough idea of possible titles (>4000 unique)
    - Reduced to [a-z] plus space, stop words removed (further removed from stop words, roman numerals added), words stemmed, ignore headings with line breaks, no more than five words
    - conclus, conclus futur, conclus further, conclus relat, discuss conclus, conclus open, futur work, chapter conclus, conclus futur work, conclus futur research, conclus perspect, conclusion
  - For each Iterate over zones until a “conclusion” heading found
    - Grab sentences until sentence starts with appendi, refer or acknowledg or ZONE is GEN_REFERENCES
  - Generate conclusions_with_fw.xml – is_fw=“false”, processed=“false”

Data Construction – Manual Annotation

- Annotation by hand using Notepad++ on Windows – loads large files
- Initially by hand – is-fw=“true”, processed=“true”
- Regular expression search to find possible further work
  - <sentence(.*)is-fw="False">(.*)(
  - ((promising|further|future) (work|research|direction|development|improve(d|ment)|stud|perspective|topic|optimi[sz]ed|optimi[sz]ation)(y|s|es|ies)?)
  - |will be interesting
  - |need to (extend|explore)
  - |(remaining|open) (research)?(question|challenge|problem)(s)?
  - |research direction
  - |((aim|going|plan|intend) (at|to)))
- Metrics.py monitored progress
- Manually placed all XML with fifteen sentences or less into the conclusions_with_fw_15_lines_or_less.xml file.

Data Construction – Post annotation clean up

- Noticed some sentences were broken up into multiple sentences
- Abbreviations - 'i.e.', 'e.g.', 'et al.', 'w.r.t.', 'cf.', 'pp.', 'etc.' and 'i.i.d.'
- Numeric lists - '1.', '2.', '3.', '4.', '5.', 'i.', 'ii.', 'iii.', 'iv.', 'v.'
- Created a script that looks for sentences ending in these tokens
  - Join with the next sentence
  - If any of the sentences are is-fw="true" then resulting sentence is-fw="true"

Data Construction – Some Dataset Statistics

	Not Further Work (85%)	Is Further Work (15%)	Totals
Training (70%)	16,005	2,344	18,349
Test (30%)	6,817	1,048	7,865
Totals	22,822	3,392	26,214

Total number of docs	3,531	Total number of sentences	26,214
Total words in corpus	628,601	Total words in corpus vocabulary	26,541
Total words in class is-fw	83,578	Total words in class is-fw vocabulary	7,642
Total words in class is-not-fw	545,023	Total words in class is-not-fw vocabulary	25,142
Words only in is-fw sentences	1,399	Words only in is-not-fw sentences	18,899

Model Construction – XML to DataFrame

- Pandas DataFrame - SQL like access, statistics, filtering, grouping
  - Can incrementally add columns as the requirements evolve
- Xml nodes/attributes to columns
- Additional columns
  - clean
  - clean_pos_tags
  - is_train
  - is_fw
  - prev_fw
  - is_question
  - is_continuation

Model Construction – Bag of words

- sentences = [‘How are you doing?’, ‘I am doing quite well, how are you doing?’]
- vocabulary = {'am': 0, 'are': 1, 'doing': 2, 'how': 3, 'quite': 4, 'well': 5, 'you': 6}

Vector Index	0	1	2	3	4	5	6
Word	am	are	doing	how	quite	well	you
Sentence 1 Count	0	1	1	1	0	0	1
Sentence 2 Count	1	1	2	1	1	1	1

Model Construction – TF-IDF

$Term\ frequency = tf(t,d)$ Number of times term, t, appears in document, d.

$Document\ frequency = df(d,t)$ Number of docs containing at least one occurrence of term, t.

n_d Total number of documents in the corpus

$$idf(t) = \ln\left(\frac{1+n_d}{1+df(d,t)}\right)+1$$ The inverse document frequency

$$tf\text{-}idf(t,d) = tf(t,d) \times idf(t)$$

Vector Index	0	1	2	3	4	5	6
Word	am	are	doing	how	quite	well	you
Sentence 1 tf(t,d)	0	1	1	1	0	0	1
Sentence 1 tf(t,d)	1	1	2	1	1	1	1
idf(t)	1.405	1	1	1	1.405	1.405	1
Sentence 1 tf(t,d) x idf(t)	0	1	1	1	0	0	1
Sentence 2 tf(t,d) x idf(t)	1.405	1	2	1	1.405	1.405	1
Sentence 1 TF-IDF (L2)	0	0.5	0.5	0.5	0	0	0.5
Sentence 2 TF-IDF (L2)	0.391	0.278	0.556	0.278	0.391	0.391	0.278

Model Construction – POS Tags

- Using Stanford NLP POS tagger wrapped by python NLTK Library
  - Their readme recommends the english-bidirectional-distsim.tagger model file
    - Best performance but slower
- [[('How', 'WRB'), ('are', 'VBP'), ('you', 'PRP'), ('doing?', 'VBP')], [('I', 'PRP'), ('am', 'VBP'), ('doing', 'VBG'), ('quite', 'RB'), ('well,', 'JJ'), ('how', 'WRB'), ('are', 'VBP'), ('you', 'PRP'), ('doing?', 'VBP')]]
- JJ – Adjective, PRP – Personal pronoun, RB – Adverb, VBG - Verb, gerund or present participle, VBP - Verb, non-3rd person singular present, WRB - Wh-adverb (e.g. where, why, when, how)
- Both bag of words and TF-IDF

Model Construction – NBSVM

- NBSVM combination of MNB and SVM – Wang and Manning (2013)
  - counts of each term in the positive class sentences
    - $\boldsymbol{p} = \alpha + \sum n_{t,pos}$
  - counts of each term in the positive class sentences
    - $\boldsymbol{q} = \alpha + \sum n_{t,neg}$
  - log count ratio
    - $\boldsymbol{r} = \ln\left(\frac{\boldsymbol{p}/\|\boldsymbol{p}\|_1}{\boldsymbol{q}/\|\boldsymbol{q}\|_1}\right)$
- Use log count ratio to train an SVM

Model Construction – Mean Word2Vec

- The average of Word2Vec with 300 dimensions for each word in sentence. (Similar to Zhang and Wallace (2015))
- If a word in the test sentence does not exist the word is ignored.
- Vector values are normalized to be between 0 and 1
  - MNB expects positive values

Model Construction – Custom Features

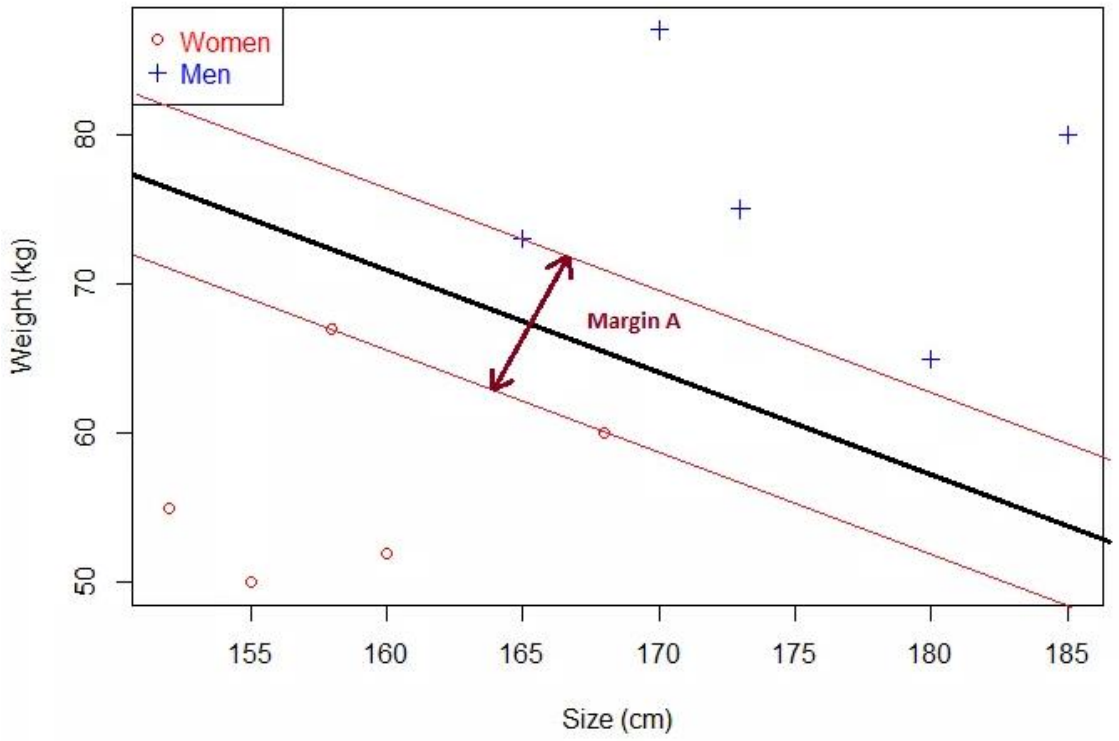
- Previous sentence is further work
- Sentence is a question
  - Noticed whilst manually annotating
- Sentence is a continuation
  - Noticed whilst manually annotating
  - List of starting phrases to compare the start of the sentence to
    - 'it', 'this', 'these', 'those', 'they', 'and', 'also', 'but', 'because', 'therefore', 'thus', 'hence', 'such', 'further', 'furthermore', 'moreover', 'another', 'first', 'firstly', 'second', 'secondly', 'thirdly', 'third', 'fourth', 'fourthly', 'finally', 'for example', 'for instance', 'in particular', 'following this', 'we note though', 'then', 'it is also likely', 'on the other hand', 'in addition’
    - Clark and Divvala’s (2016) use of a list of key phrases to identify captions
- Weight these custom features
  - Improved MNB performance

Model Construction – MNB

- Multinomial Naïve Bayes
- Used in multiple research papers as a baseline model
- Bayes theorem applied to sentence classification
  - $P(y|x_1, x_2, \dots, x_n) = \frac{P(x_1,x_2,\dots,x_n|y)P(y)}{P(x_1,x_2,\dots,x_n)}$
- When comparing two class probabilities, denominator can be ignored
  - $P(y|x_1, x_2, \dots, x_n) \propto P(x_1, x_2, \dots, x_n|y)P(y)$
- Simplifies to
  - $\hat{y} = \arg\max P(y) \prod_{i=1}^{n} P(x_i|y)$

Model Construction – SVM

- SVM – Cortes and Vapnik (1992)
- Hyperplanes separating classes
  - Maximum margin
  - optimisation technique
    - e.g. stochastic gradient descent



Model Construction - CNN

- Shallow and wide CNN – Kim (2014)
- Input
  - Word2Vec matrix
  - Additional custom features
- 3 parallel convolutional layers
- Feature maps through 1-max pooling layer
- Softmax layer to classify
- Adadelta optimizer (Zeiler, 2012)
  - Zhang and Wallace (2015) determined this produce best results

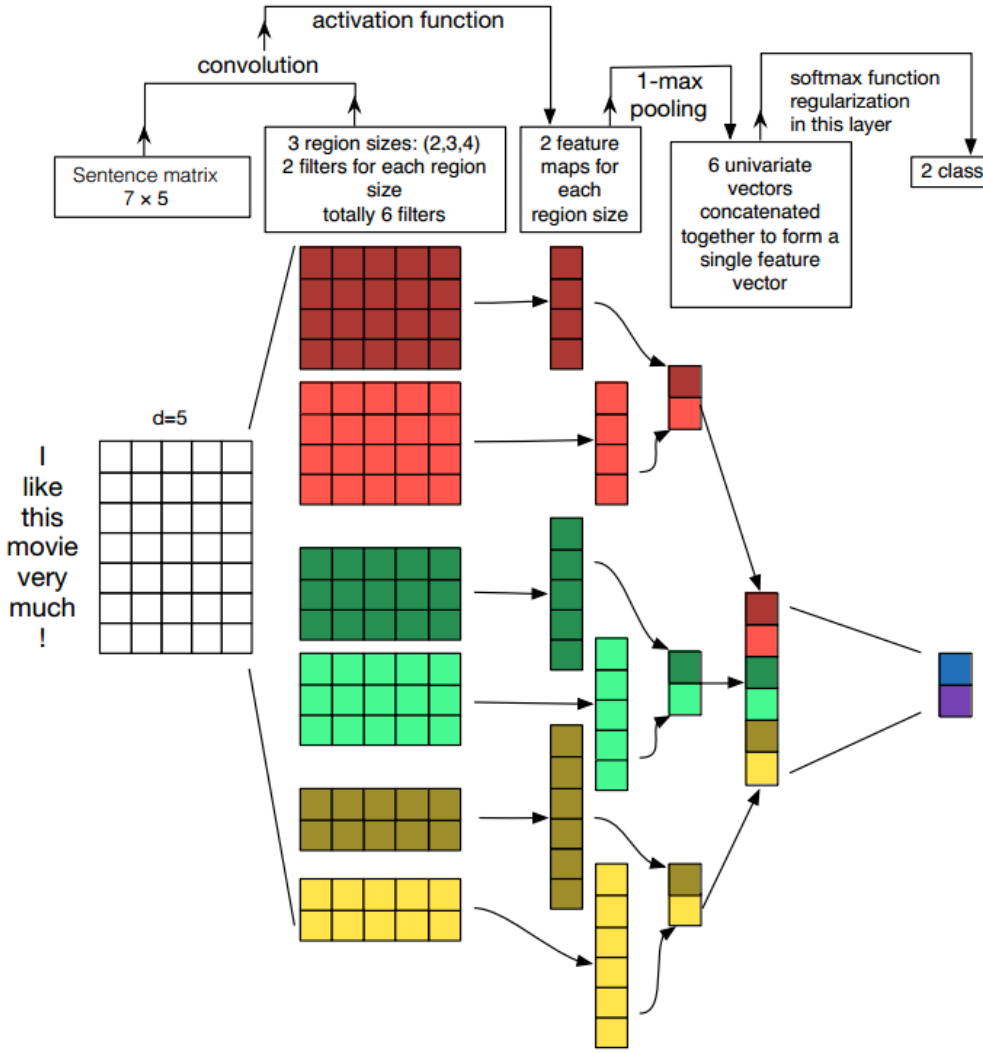


image from Zhang and Wallace (2015)

Model Construction – Unbalanced Datasets

- Random Undersampling
  - Randomly removes majority class samples - $2344 \times 2 = 4688$ samples
- Random Oversampling
  - Randomly duplicates minority class - $16005 \times 2 = 32010$ samples
- Synthetic Minority Over-Sampling Technique (SMOTE) – Chawla (2002)
  1. A sample is selected at random.
  2. The k-nearest neighbours of this sample are determined.
  3. Randomly select one of these neighbours.
  4. Creates a new sample on the line between the original sample and the selected neighbour.
  5. This synthetic sample is assigned to the minority class.
- imblearn library cannot use sparse matrices
  - impractical for large number of features
  - Only used for CNN

Model Construction - Experiments

- MNB and SVM - 2,339 and 2,489 sets of features to try.
  - Counts and TF-IDFS types, n-gram ranges and custom features
    - $10 \times 15 \times 15 = 2250$
  - NBSVM, n-gram ranges and custom features (SVM only)
    - $10 \times 15 = 150$
  - Special features only
    - 14
  - Mean word2vec sizes and custom features
    - $5 \times 15 = 75$
- CNN
  - Five features
    - Resampling – None, Random Undersampling, Random Oversampling, SMOTE
    - Maximum Sentence Length – 100, 150 or 200
    - Word2Vec dimensions – 100, 300 or 500
    - Number of filters – 100, 200 or 300
    - Use custom features – True or False
  - 3 stages
    - 2,000 steps – $4 \times 3 \times 3 \times 3 \times 2 = 216$ experiments
    - 20,000 steps – top 20 from previous stage
    - 50,000 steps – top 3 from previous stage
- Output metrics to pandas DataFrame for analysis

Analysis of models - Metrics

- $Accuracy\ (Acc) = \frac{(TP+TN)}{(TP+FP+FN+TN)}$
- $Precision = \frac{TP}{TP+FP}$
- $Recall = \frac{TP}{TP+FN}$
- $F1 - Score = \frac{2 \times Precision \times Recall}{Precision+Recall} = \frac{2TP}{(2TP+FP+FN)}$

Predicted by classifier	Actual Class	
	Positive	Negative
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

- $Accuracy\ (Acc) = \frac{(0+9900)}{(0+100+0+9900)} = \frac{9900}{10000} = 0.99$
- $Precision = \frac{0}{0+100} = 0$
- $Recall = \frac{0}{0+0} = 0$
- $F1 - Score = \frac{2 \times 0}{(2 \times 0+30+0)} = \frac{0}{30} = 0$

Predicted by classifier	Actual Class	
	Positive	Negative
Positive	0	100
Negative	0	9,900

"Useless" classifier Confusion Matrix

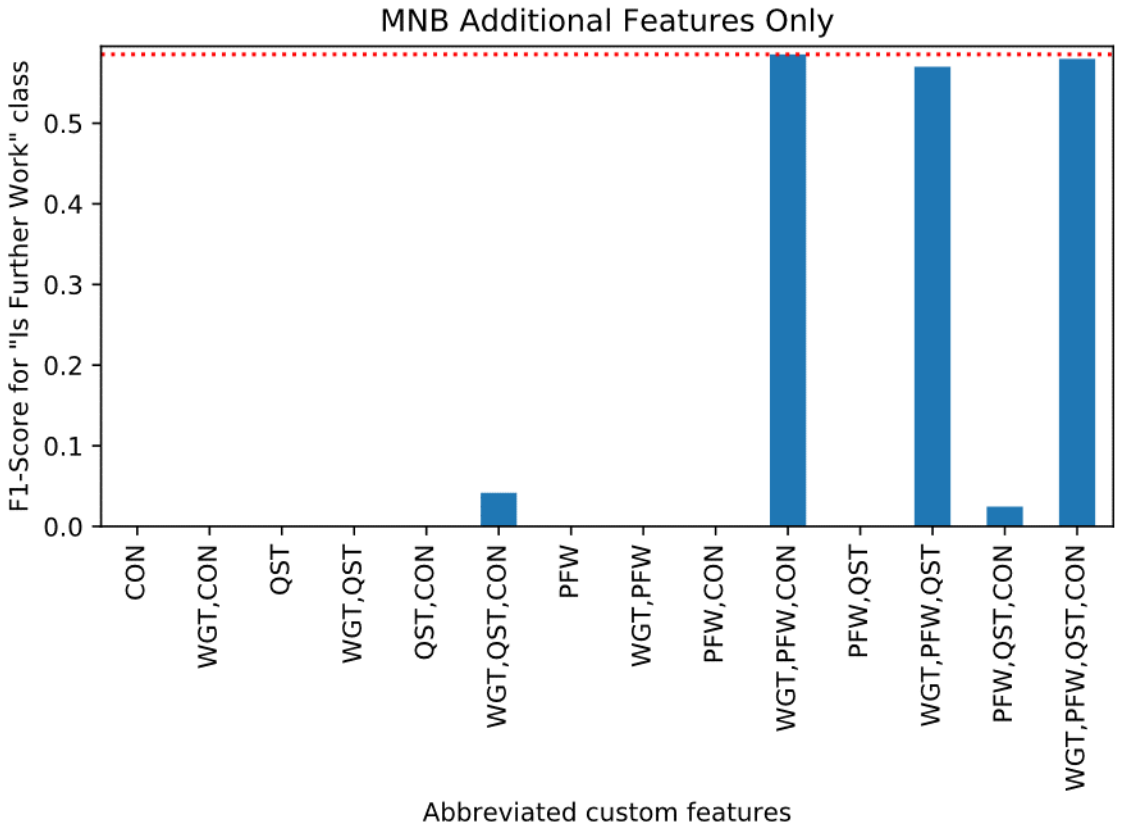
Analysis of models - Abbreviations

WGT	Weight additional features	RUS	Random Over Sampling
PFW	Previous sentence is FW	ROS	Random Over Sampling
QST	Is question sentence?	SMOTE	Synthetic Minority Over-sampling TEchnique
CON	Is continuation sentence?		
CTW	Count of words	TIW	TF-IDF of words
CTP	Count of POS tags	TIP	TF-IDF of POS Tags

Analysis of models – MNB Additional Features Only

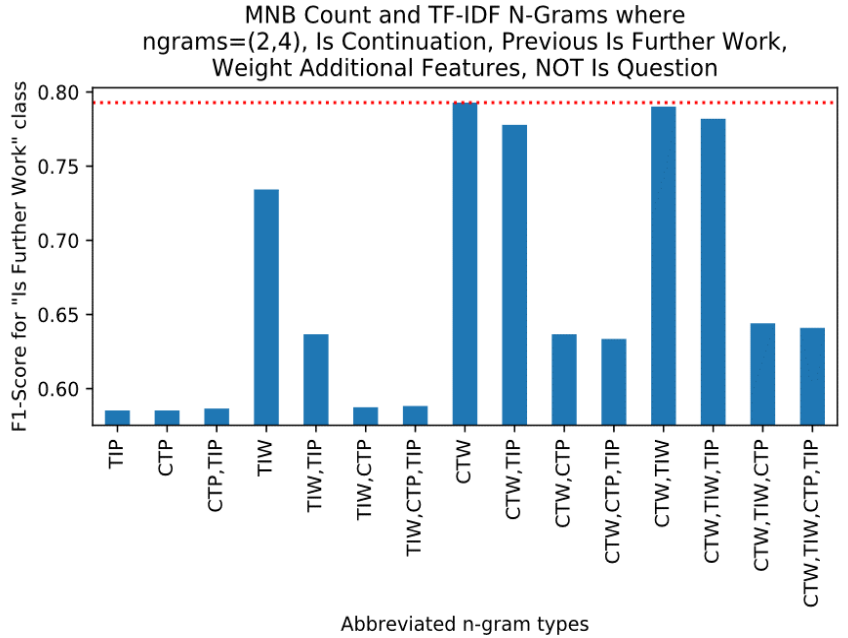
- Fourteen experiments
  - Using only the additional features
    - CON
    - PFW
    - QST
    - WGT

Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	WGT, PFW, CON
0.9236	0.7433	0.9748	0.4752	0.9485	0.5797	WGT, PFW, QST, CON
0.9210	0.7653	0.9786	0.4542	0.9489	0.5701	WGT, PFW, QST
0.8688	0.4340	0.9956	0.0219	0.9279	0.0418	WGT, QST, CON
0.8682	1.0000	1.0000	0.0124	0.9294	0.0245	PFW, QST, CON
0.8668	0.0000	1.0000	0.0000	0.9286	0.0000	NOTE: All other combinations produce these results

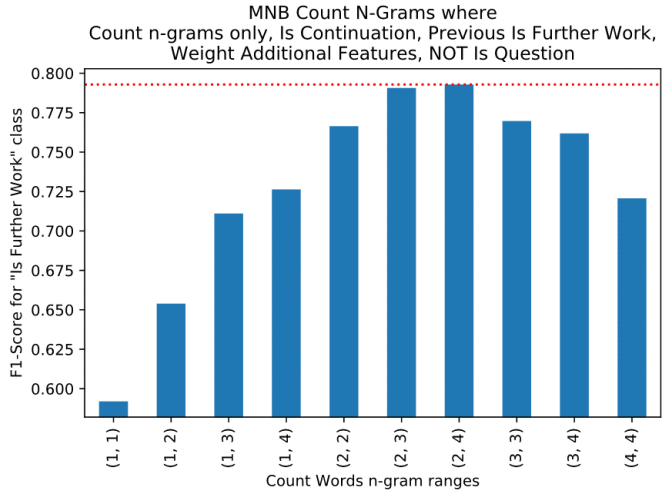


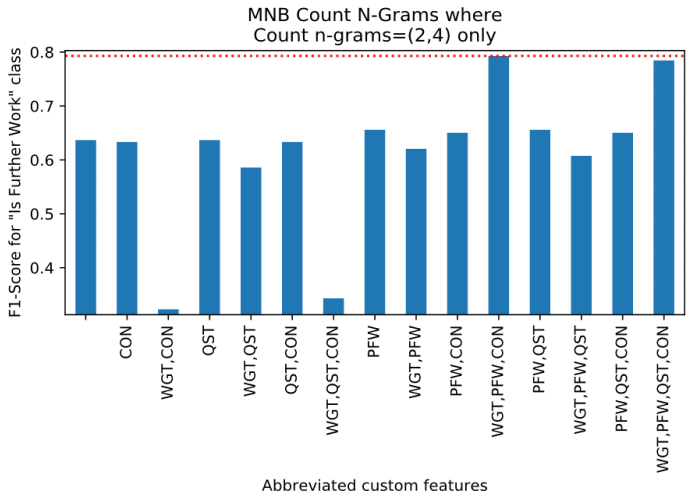
Analysis of models – MNB Counts and TF-IDF n-grams

- Two thousand two hundred and fifty experiments
  - Counts and TF-IDFS types, n-gram ranges and custom features
    - $10 \times 15 \times 15 = 2250$
  - CTW, CTP, TIW & TIP
  - (1,1), (1,2), (1,3), (1,4), (2,2), (2,3), (2,4), (3,3), (3,4), (4,4)
  - CON, PFW, QST, WGT



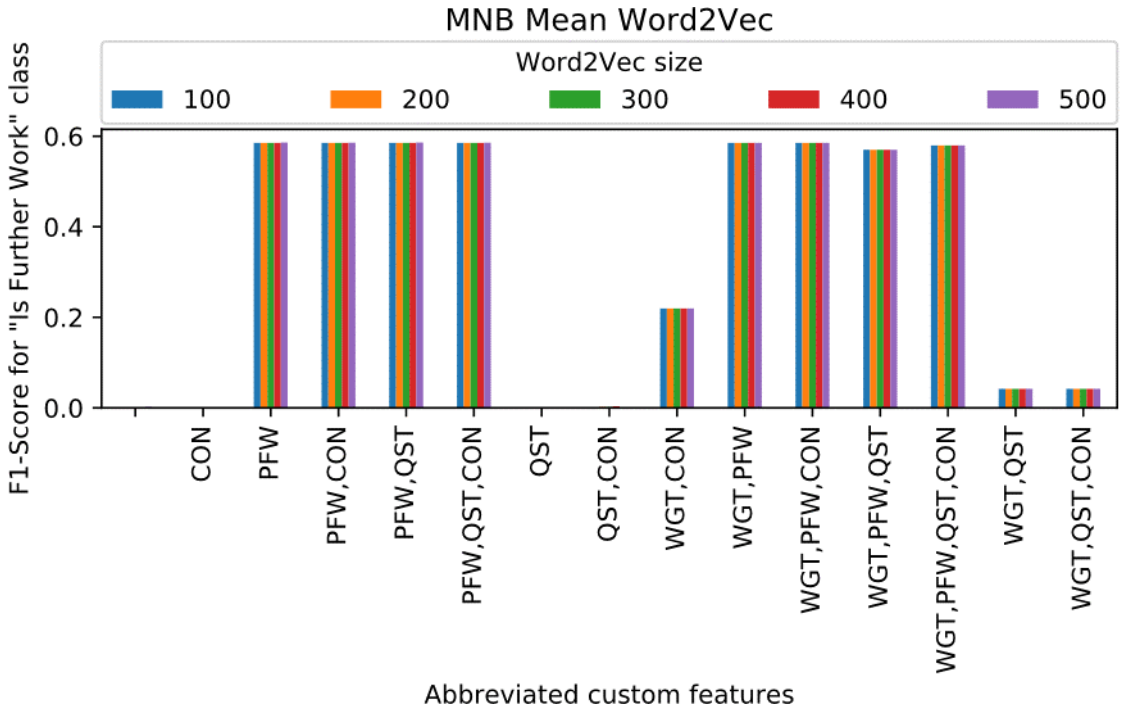
Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9668	0.8025	0.9704	0.7834	0.9686	0.7929	Count Words=(2, 4) WGT, PFW, CON
0.9677	0.7931	0.9683	0.7901	0.9680	0.7916	Count Words=(2, 3) TF-IDF Words=(2, 3) WGT, PFW, CON
0.9653	0.8092	0.9720	0.7729	0.9686	0.7906	Count Words=(2, 3) WGT, PFW, CON
0.9688	0.7828	0.9660	0.7977	0.9674	0.7902	Count Words=(2, 4) TF-IDF Words=(2, 4) WGT, PFW, CON
0.9670	0.7838	0.9667	0.7853	0.9668	0.7846	Count Words=(2, 4) WGT, PFW, QST, CON





Analysis of models – MNB Mean Word2Vec Size

- Seventy-five experiments
  - $5 \times 15 = 75$
  - Sizes – 100, 200, 300, 400, 500
  - Custom Features
    - CON
    - PFW
    - QST
    - WGT

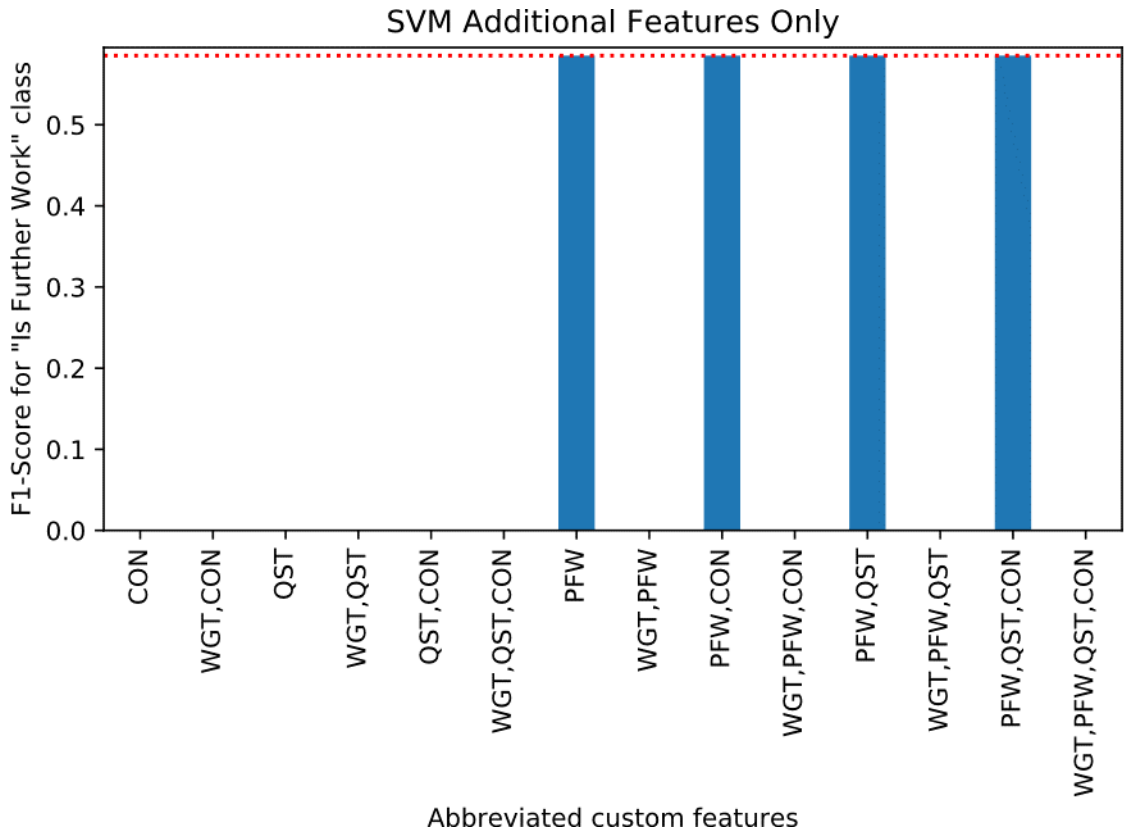


Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9233	0.7743	0.9789	0.4714	0.9503	0.5860	Mean Word2Vec size=500 PFW
0.9233	0.7743	0.9789	0.4714	0.9503	0.5860	Mean Word2Vec size=500 PFW, QST
0.9233	0.7731	0.9787	0.4714	0.9502	0.5857	Mean Word2Vec size=500 PFW, CON
0.9233	0.7731	0.9787	0.4714	0.9502	0.5857	Mean Word2Vec size=500 PFW, QST, CON
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	Mean Word2Vec size=500 PFW

Analysis of models – SVM Additional Features Only

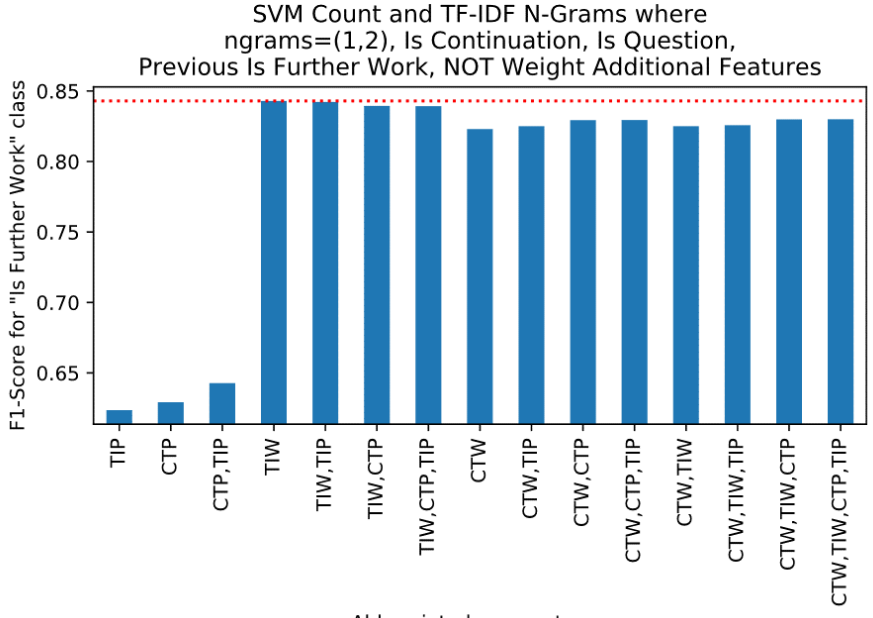
- Fourteen experiments
  - Using only the additional features
    - CON
    - PFW
    - QST
    - WGT

Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	PFW
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	PFW, CON
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	PFW, QST
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	PFW, QST, CON
0.8668	0.0000	1.0000	0.0000	0.9286	0.0000	NOTE: All other combinations produce these results

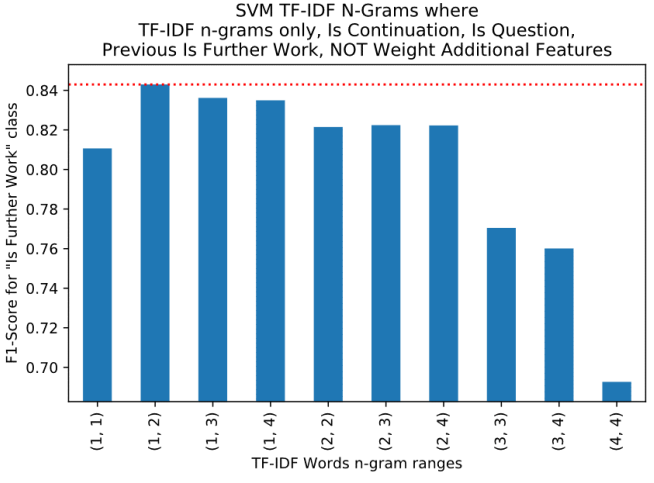


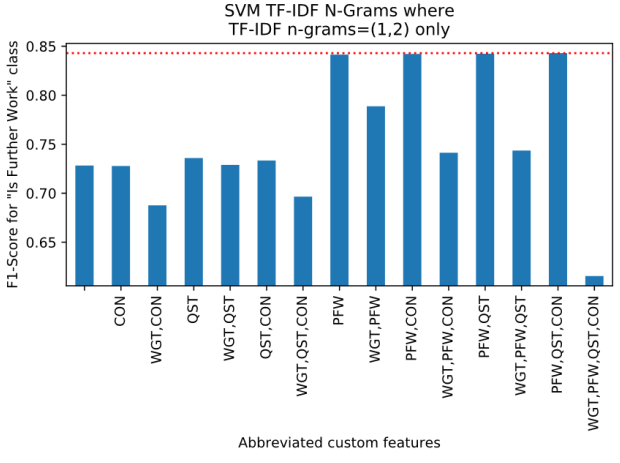
Analysis of models – SVM Counts and TF-IDF n-grams

- Two thousand two hundred and fifty experiments
  - Counts and TF-IDFS types, n-gram ranges and custom features
    - $10 \times 15 \times 15 = 2250$
  - CTW, CTP, TIW & TIP
  - (1,1), (1,2), (1,3), (1,4), (2,2), (2,3), (2,4), (3,3), (3,4), (4,4)
  - CON, PFW, QST, WGT



Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9758	0.8434	0.9759	0.8426	0.9759	0.8430	TF-IDF Words=(1, 2) PFW, QST, CON
0.9747	0.8503	0.9774	0.8349	0.9760	0.8426	TF-IDF Words=(1, 2) TF-IDF POS tags=(1, 2) PFW, QST
0.9744	0.8517	0.9777	0.8330	0.9761	0.8423	TF-IDF Words=(1, 2) TF-IDF POS tags=(1, 2) PFW, QST, CON
0.9755	0.8439	0.9761	0.8406	0.9758	0.8423	TF-IDF Words=(1, 2) PFW, QST
0.9758	0.8418	0.9756	0.8426	0.9757	0.8422	TF-IDF Words=(1, 2) PFW, CON

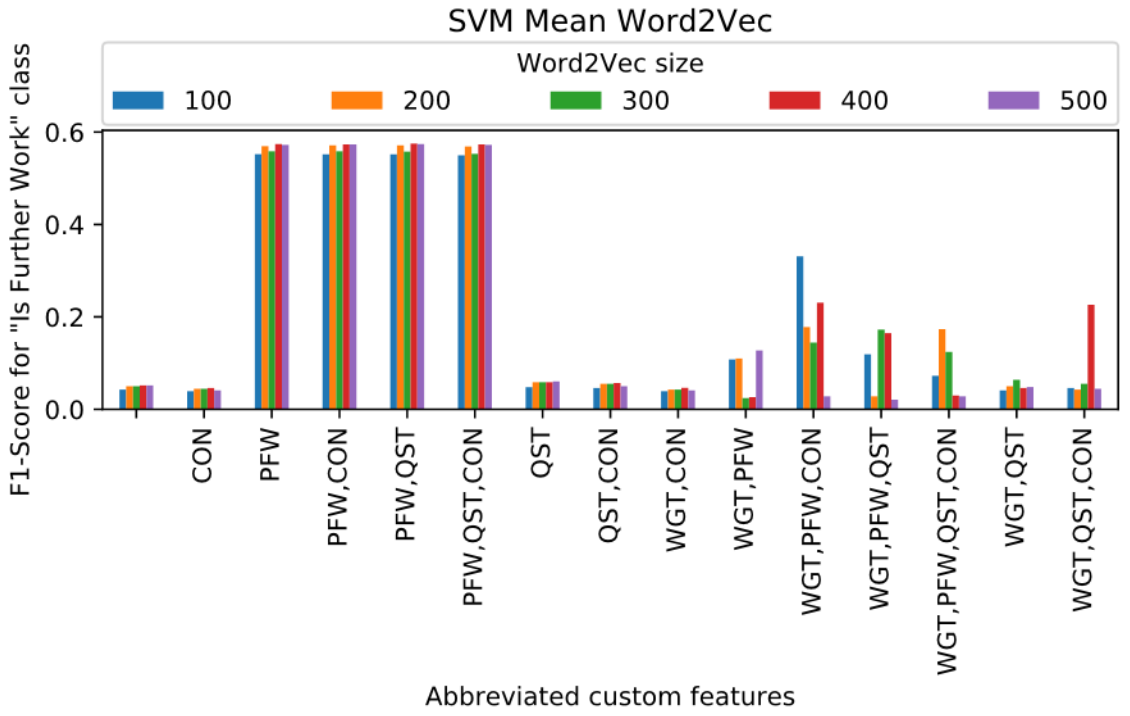




Analysis of models – SVM Mean Word2Vec Size

- Seventy experiments
  - $5 \times 15 = 75$
  - Sizes – 100, 200, 300, 400, 500
  - Custom Features
    - CON
    - PFW
    - QST
    - WGT

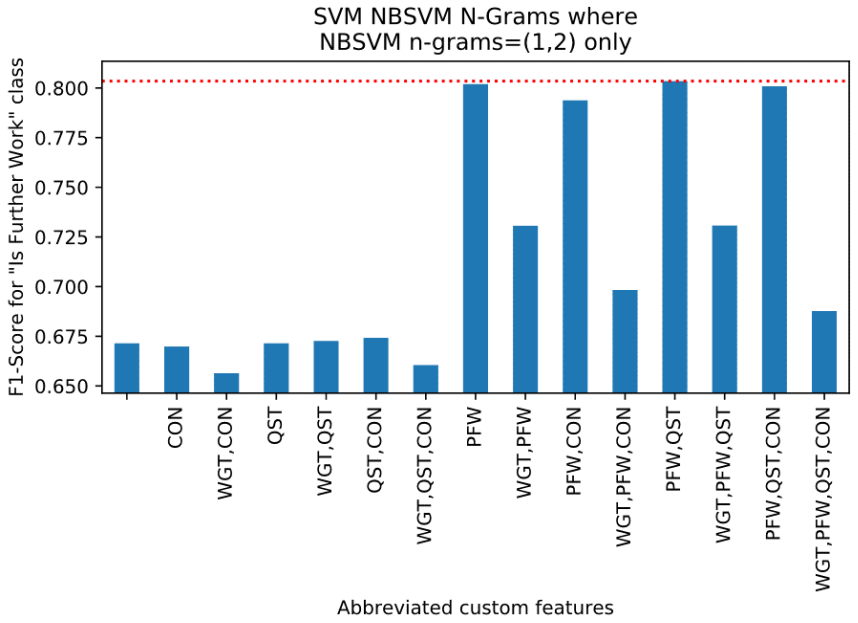
Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9214	0.7772	0.9799	0.4561	0.9497	0.5749	Mean Word2Vec size=400 PFW, QST
0.9213	0.7769	0.9799	0.4552	0.9497	0.5740	Mean Word2Vec size=400 PFW
0.9213	0.7769	0.9799	0.4552	0.9497	0.5740	Mean Word2Vec size=500 PFW, QST
0.9211	0.7765	0.9799	0.4542	0.9496	0.5731	Mean Word2Vec size=400 PFW, CON
0.9211	0.7765	0.9799	0.4542	0.9496	0.5731	Mean Word2Vec size=400 PFW, QST, CON

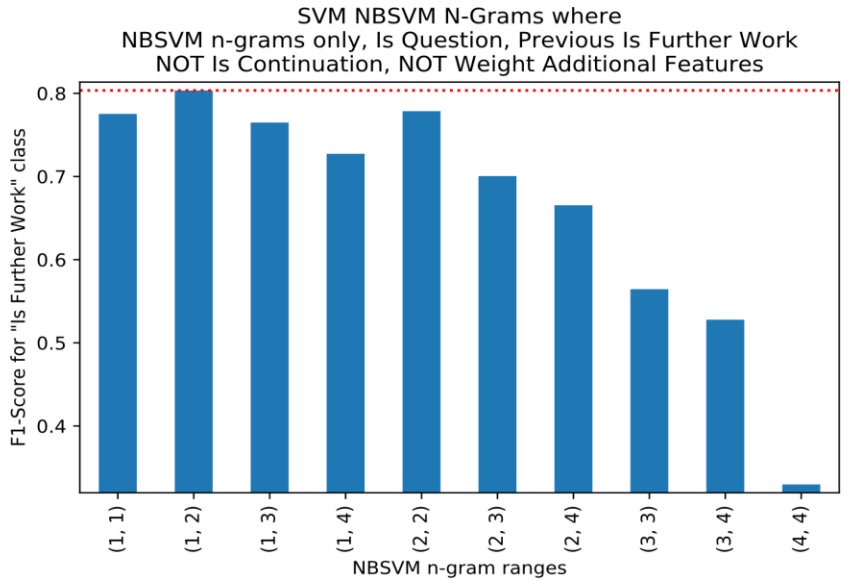


Analysis of models – SVM NBSVM

- One-hundred and fifty experiments
  - $10 \times 15 = 150$
  - (1,1), (1,2), (1,3), (1,4), (2,2), (2,3), (2,4), (3,3), (3,4), (4,4)
  - CON, PFW, QST, WGT

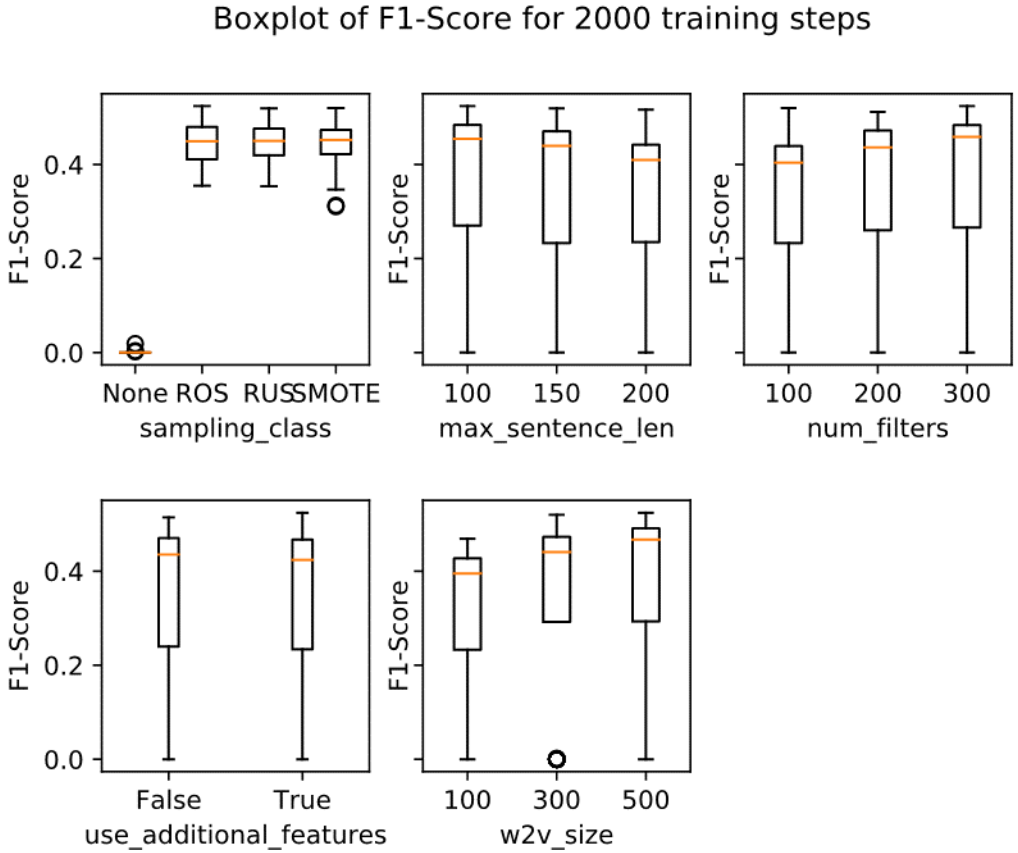
Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9569	0.9254	0.9912	0.7099	0.9738	0.8035	NBSVM=(1, 2) PFW, QST
0.9565	0.9263	0.9913	0.7071	0.9736	0.8019	NBSVM=(1, 2) PFW
0.9569	0.9185	0.9903	0.7099	0.9733	0.8009	NBSVM=(1, 2) PFW, QST, CON
0.9557	0.9142	0.9899	0.7013	0.9725	0.7937	NBSVM=(1, 2) PFW, CON
0.9506	0.9405	0.9935	0.6641	0.9716	0.7785	NBSVM=(2, 2) PFW, QST





Analysis of models – CNN 2,000 training steps

- Two-hundred and sixteen experiments
  - $4 \times 3 \times 3 \times 3 \times 2 = 216$
- Features
  - Sampling = None, ROS, RUS, SMOTE
  - Max. Sentence Length = 100, 150, 200
  - Number of filters = 100, 200, 300
  - Use additional features = False, True
  - Word2Vec size = 100, 300, 500

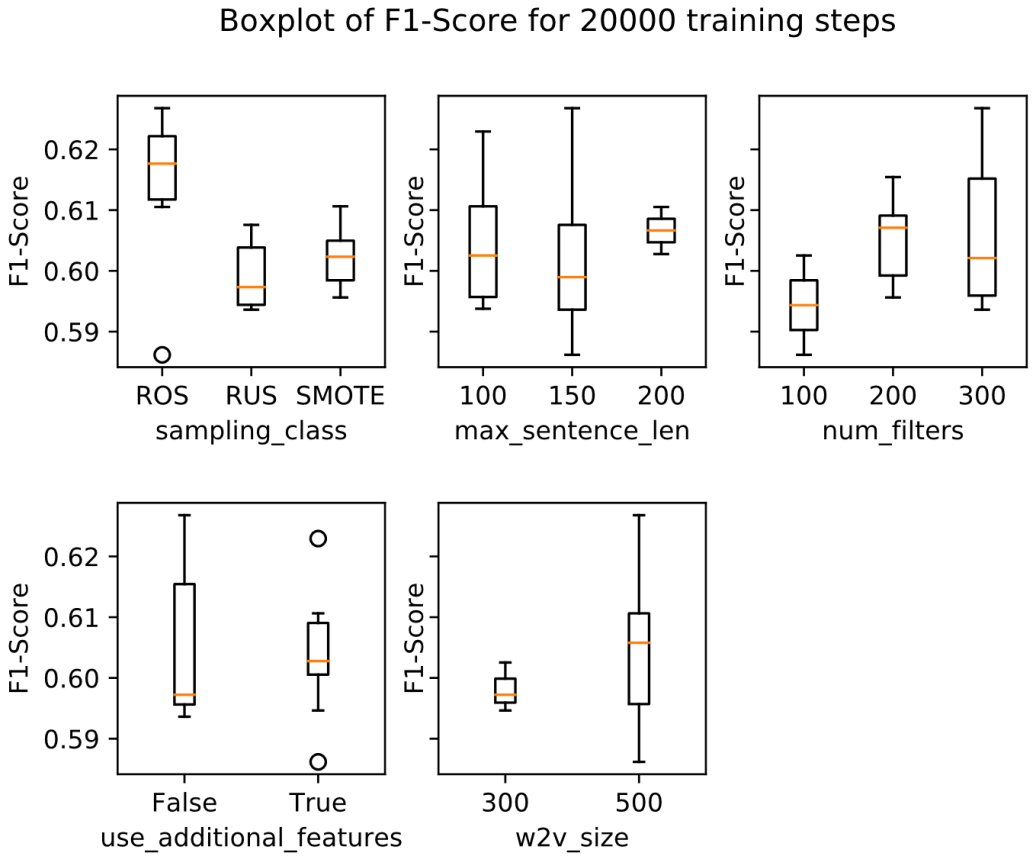


Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	Max. sentence length, Number of filters, Word2Vec size, Additional Features, Sampling Type
0.9577	0.3992	0.8234	0.7634	0.8855	0.5242	100, 300, 500, Yes, ROS
0.9448	0.4233	0.8589	0.6737	0.8998	0.5199	100, 100, 300, Yes, SMOTE
0.9569	0.3946	0.8209	0.7595	0.8837	0.5194	150, 300, 500, Yes, RUS
0.9557	0.3936	0.8219	0.7519	0.8838	0.5167	200, 300, 500, Yes, ROS
0.9537	0.3944	0.8254	0.7395	0.8850	0.5144	100, 300, 500, No, ROS

Analysis of models – CNN 20,000 training steps

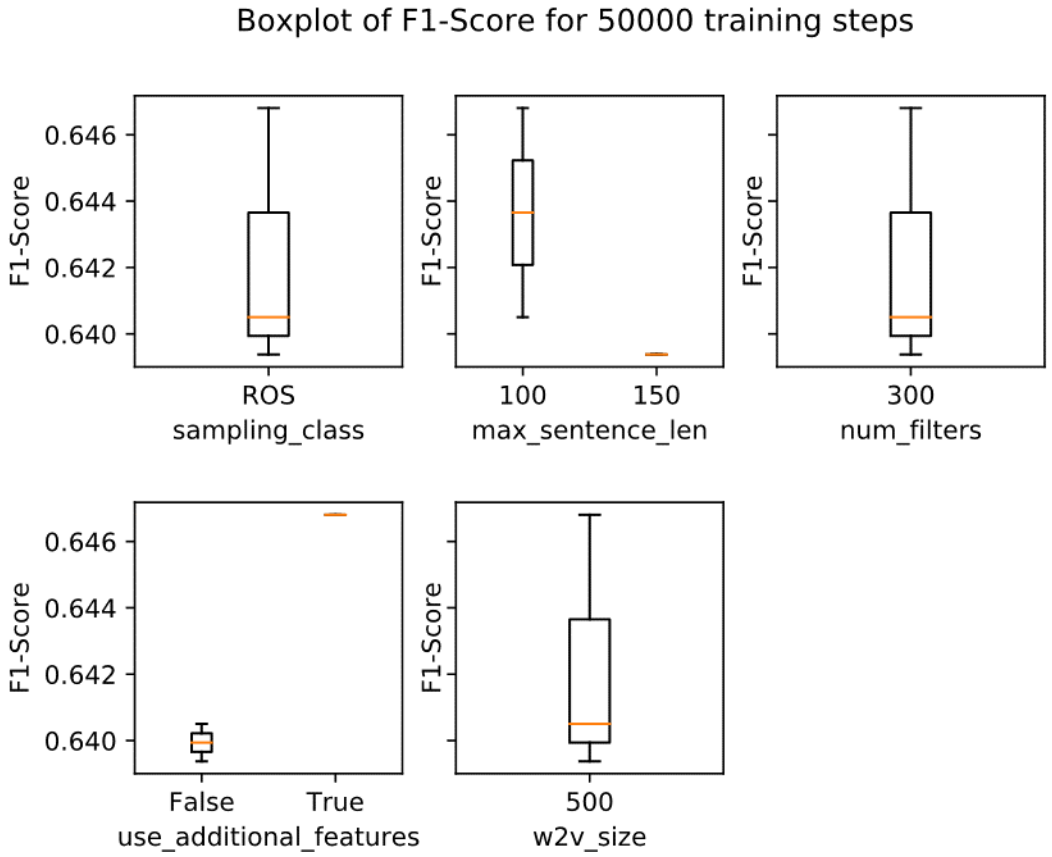
- Top twenty from previous stage

Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	Max. sentence length, Number of filters, Word2Vec size, Additional Features, Sampling Type
0.9627	0.5262	0.8928	0.7748	0.9264	0.6268	150, 300, 500, No, ROS
0.9626	0.5208	0.8904	0.7748	0.9251	0.6229	100, 300, 500, Yes, ROS
0.9619	0.5183	0.8898	0.7710	0.9245	0.6199	100, 300, 500, No, ROS
0.9620	0.5117	0.8868	0.7719	0.9228	0.6154	100, 200, 500, No, ROS
0.9527	0.5373	0.9064	0.7071	0.9290	0.6106	100, 200, 500, Yes, SMOTE



Analysis of models – CNN 50,000 training steps

- Top three results from previous stage



Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	Max. sentence length, Number of filters, Word2Vec size, Additional Features, Sampling Type
0.9649	0.5489	0.9005	0.7872	0.9316	0.6468	100, 300, 500, Yes, ROS
0.9636	0.5440	0.8997	0.7786	0.9305	0.6405	100, 300, 500, No, ROS
0.9641	0.5405	0.8978	0.7824	0.9297	0.6394	150, 300, 500, No, ROS

Conclusions – Comparing the best of each batch

- Worst models
  - F1-Score = 0.5749 to 0.5860
  - Mean Word2Vec Size
  - Additional custom features only
- In the middle
  - CNN
  - F1-Score = 0.6468
- Best models
  - F1-Score = 0.7929 to 0.8430
  - Count/TF-IDF models
  - NBSVM

Precision		Recall		F1-Score		Features
Not FW	FW	Not FW	FW	Not FW	FW	
0.9758	0.8434	0.9759	0.8426	0.9759	0.8430	SVM: TF-IDF Words=(1, 2) PFW, QST, CON
0.9569	0.9254	0.9912	0.7099	0.9738	0.8035	SVM: NBSVM=(1, 2) PFW, QST
0.9668	0.8025	0.9704	0.7834	0.9686	0.7929	MNB: Count Words=(2, 4) WGT, PFW, CON
0.9649	0.5489	0.9005	0.7872	0.9316	0.6468	CNN: Max. sentence length = 100, Number of filters = 300, Word2Vec size = 500, Additional Features = Yes, Sampling Type = ROS
0.9233	0.7743	0.9789	0.4714	0.9503	0.5860	MNB: Mean Word2Vec size=500 PFW
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	MNB: WGT, PFW, CON
0.9233	0.7719	0.9786	0.4714	0.9502	0.5853	SVM: PFW
0.9214	0.7772	0.9799	0.4561	0.9497	0.5749	SVM: Mean Word2Vec size=400 PFW, QST

Conclusions – The best model

- Best model
  - A Linear Support Vector Classifier
  - TF-IDF of the unigrams and bigrams of sentence words
  - Is the previous sentence a further work sentence?
  - Is the sentence a question?
  - Is the sentence a continuation of the thread of the previous sentence?
- F1-Scores 0.8430 “is further work” and 0.9759 “is not further work”
- Correctly predicts (Recall)
  - 883 of 1048 (0.8426) – “is further work”
  - 6653 of 6817 (0.9759) – “is not further work”

Predicted by classifier	Actual Class	
	Is FW	Not FW
Is FW	883	165
Not FW	164	6653

Conclusions – Incorrectly classified sentences

- 329 test sentences are incorrectly classified.
  - 164 which should be “is FW”.
  - 165 which should be “not FW”.
- all should be further work
  - Manual Annotation errors
    - "**In the future**, **we would also like** to apply this technique for other practical problems arising biometrics vision speech processing etc also there has been a lot of work on supervised dictionary learning our preliminary formulation is unsupervised."
    - "**In future**, **we expect to improve** the results even further by incorporating techniques from supervised learning.“
    - "**Further work** on this problem **will improve** the ability to use isolated instances that do not have a corresponding multi-view representation to improve learning and enable multi-view learning to be used for a wider variety of applications."
  - "An **open question** on these iterations is which valid formulas are realizable.”

Conclusions – Dataset Construction

- Time consuming and potentially error prone
- arXiv bulk download did not contain all PDFs
- CERMINE PDF extractor
  - Batch process that took a few seconds per paper
  - Not a perfect extraction
    - Missing letters, words concatenated, section headers containing body text
    - Couldn’t always identify conclusion/further work sections
    - Issues likely to reduce performance of models
- Manual annotation is error prone – humans get tired and distracted
  - Notepad++ made the process as painless as possible – large text files and regular expression searching
- Sentence tokenization issues
  - Incorrect interpretation of full stop
- NLTK and Stanford NLP – excellent libraries with excellent documentation

Conclusions – Model Construction

- Libraries - Pandas, lxml, BeautifulSoup, NumPy, SciPy, sci-kit learn, TensorFlow, gensim and imblearn.
  - Great documentation and intuitive APIs
- MNB and SVM
  - Trained in seconds
  - Predict very quickly.
  - MNB trains and predicts quicker than SVM
  - MNB could not handle negative features
  - MNB performed better with weighted custom features
    - Exception mean Word2Vec – probably due to 0 to 1 scaling
  - Large feature sets, between 18,349 and 1,743,115, benefitted from sparse matrices (SciPy)
    - Imblearn uses dense matrices and could not cope with resampling large feature sets
- CNN
  - Did not perform as well as the best MNB and SVM models
  - May be due to small training set
  - Resampling with imblearn was possible due to much smaller feature set

Further work – Dataset construction

- Data Acquisition
  - Structured research paper format, e.g. NLM JATS. for algorithmic processing
    - Journal Article Tag Suite – National Institute of Health
    - Biomedical field
    - CERMINE supports extraction to this format
- Data Cleaning
  - Spelling mistakes – spell checker or similarity measures
    - Specialist words may require specialist dictionaries
  - Acronym expansion
  - Use of dictionary and character or word look ahead to solve:
    - Broken words – e.g. “probabilist” “ic”
    - Concatenated words – e.g. “establishedunderwhich”
    - Problems with “real” words made up of two words e.g. ahead, a-head

Further work – Dataset construction

- Data Cleaning (continued)
  - Table and equation extraction/identification
  - “Is a continuation sentence” feature
    - Currently uses lookup – could be machine learning trained on a dataset
  - “Is a question” feature
    - Currently looks for “?”
    - What, why, where, how, etc.
    - Could be machine learning trained on a dataset
  - Handling foreign languages
  - Better sentence tokenization
- Larger training/test dataset
  - Manually annotate more data
  - Semi-supervised learning such as self training
    - Use a weak classifier to label un-labelled data and then retrain the classifier to make stronger.
    - Xuan et al. (2017) uses this approach.
  - Generative models
    - Kingma et al. (2014) create a deep learning model to generate new training data from existing supervised data.

Further work – Model Construction

- Issue with “Previous Sentence is further work” feature
  - Document based processing, sentence by sentence, then status of previous sentence is known
  - If “Previous Sentence is further work” is not identified then perhaps subsequent sentences will not be identified.
  - Current tests are in isolation so have not examined this issue
- Removing irrelevant words to reduce feature dimensions
  - Currently “all” words are used but not all are likely to be relevant
  - Further analysis of sentences could identify relevant words
- Resampling
  - MNB and SVM models did not use resampling due to imblearn memory issues
  - Imblearn is open source, so could perhaps be modified to use SciPy sparse matrices
- Deep Learning
  - TensorFlow has support for word embeddings – GPU speed improvements
  - CNN character level models (Zhang, Zhao and LeCun, 2015)
  - LSTM approaches (Liu, Qiu and Huang, 2016)

Further work – Productionizing the model

- Helping researchers do there work
- Convert the existing code to an application or callable API
- Test the model on new research papers
  - Test a paper in situ rather than isolated sentences
    - Conclusion/Further work
    - Entire paper
  - Other subject areas
- Text summarization
  - Could be a lot of further work sentences
  - Summarization for display on a research paper summary page

References

BOUREAU, Y., PONCE, J. & LECUN, Y., (2010) A theoretical analysis of feature pooling in visual recognition. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. pp. 111–118.

CLARK, C. and DIVVALA, S., (2015) Looking Beyond Text: Extracting Figures, Tables and Captions from Computer Science Papers. *AAAI Workshops, Austin, January 2015.* California: AAAI, pp. 2-8

CORNELL UNIVERSITY LIBRARY (2017) *arXiv.org e-Print archive*. [Online] The Apache Software Foundation. Available from: https://arxiv.org/ [Accessed 08/08/2017]

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

ELMAN, J.L., (1990) Finding structure in time. *Cognitive science*, 14(2) pp.179–211.

GOOGLE, (2017) *Google Scholar*. [Online] Google. Available from: https://scholar.google.co.uk/ [Accessed 03/09/2017]

HOCHREITER, S. & SCHMIDHUBER, J., (1997) Long short-term memory. *Neural computation*, 9(8) pp.1735–1780.

JAIN, S. et al., (2015) Weakly supervised learning of biomedical information extraction from curated data. *BMC bioinformatics,* 17 (Suppl 1), pp. 1-13.

KESHAV, S., (2007) How to Read a Paper. *SIGCOMM Comput. Commun. Rev.*, 37(3) pp.83–84.

KIM, S.N. et al., (2013) Automatic keyphrase extraction from scientific articles. *Language Resources and Evaluation*, 47(3) pp.723–742.

KINGMA, D.P. et al., (2014) Semi-supervised learning with deep generative models. In: *Advances in Neural Information Processing Systems*. Advances in Neural Information Processing Systems. pp. 3581–3589.

LE, H.T., CERISARA, C. & DENIS, A., (2017) Do Convolutional Networks need to be Deep for Text Classification? *arXiv preprint arXiv:1707.04108*.

References (continued)

LIU, P., QIU, X. & HUANG, X., (2016) Recurrent neural network for text classification with multi-task learning. *arXiv preprint arXiv:1605.05101*.

LLORET, E., ROMÁ-FERRI, M.T. and PALOMAR, M., (2013) COMPENDIUM: A text summarization system for generating abstracts of research papers. *Data & Knowledge Engineering*, 88 pp.164–175.

MIKOLOV, T. et al., (2013) Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

POPOVA, S. and DANILOVA, V., (2014) Keyphrase Extraction Abstracts Instead of Full Papers. In: *2014 25th International Workshop on Database and Expert Systems Applications*. pp. 241–245.

SONG, M. et al., (2015) PKDE4J: Entity and relation extraction for public knowledge discovery. *Journal of Biomedical Informatics*, 57, pp.320–332.

TKACZYK, D. et al., (2015) CERMINE: automatic extraction of structured metadata from scientific literature. *International Journal on Document Analysis and Recognition (IJDAR)*, 18 (4) pp.317–335.

WANG, S. and MANNING, C.D., (2012) Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*. Jeju Island, Korea: Association for Computational Linguistics, pp. 90–94.

XUAN, J. et al., (2017) Automatic bug triage using semi-supervised text classification. *arXiv preprint arXiv:1704.04769*.

YOGATAMA, D. et al., (2017) Generative and discriminative text classification with recurrent neural networks. *arXiv preprint arXiv:1703.01898*.

ZHANG, Y. and WALLACE, B., (2015) A sensitivity analysis of (and practitionerś guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.

ZHANG, X., ZHAO, J. & LECUN, Y., (2015) Character-level convolutional networks for text classification. In: *Advances in neural information processing systems*. pp. 649–657.

Thank you for your time.

Are there any questions?