Lazy Prices Reprised: Vector Representations of Financial Disclosures and Market Out-performance

Kuspa Kai
Department of Computer Science
Stanford University
kuspakai@stanford.edu

Victor Cheung
Department of Computer Science
Stanford University
hoche@stanford.edu

Alex Lin
Department of Computer Science
Stanford University
alin719@stanford.edu

Abstract

The ”Efficient Market Hypothesis” (EMH) states that market out-performance is impossible through expert selection because each stock price efficiently incorporates and reflects all relevant evaluative information. We explore the validity of EMH by analyzing the latent information of financial disclosures year over year. Specifically, we explore the concept of ”Lazy Prices”, the idea that changes in financial disclosures are correlated with a decrease in market capitalization, using natural language processing methods to factor in these changes the market may not capture. We created a novel database of financial disclosures represented as GloVe vectors from 60,000 raw 10-K documents filed with the Securities and Exchange Commission (SEC) from 1994-2016, and trained several models to predict future market performance. When initially trained on the S&P 500 our model could not predict better than random; we then expanded our data set by 30x. Our best model now achieves predictive accuracy greater than 56% on the test set. We present our dataset, methodology for latent information mining, and results as well as a discussion of future improvements.

1 Introduction and Related Work

“Lazy Prices” (Cohen, 2010) found that firms that modified their periodic financial reports rather than defaulting to boilerplate tended to perform worse in the future compared to firms that did not modify their disclosures. This indicates the existence of abnormal returns. For example, suppose a company changes their annual 10-K disclosure by inserting a sentence into a section describing risk factors. Knowing which particular risk factor was added is not necessary for evaluating market performance in this case, because the relevant feature is the implicit information that risk has changed. The measures used by Cohen et. al. were TF-IDF and other string edit distances. Cohen et. al. used the magnitude of edit distances between as a scale for portfolio management, buying ”non-changers” and shorting ”changers”. Using this method, they achieved a rate of return of 30-60 basis points month over month over the following year.

Of particular interest is the possibility that more sophisticated parsing and representation of documents may better capture latent information of the exact changes that lead to financial out-performance. Finding methods that capture semantic meaning or hierarchical structure in changes to these financial disclosures that are otherwise obscure to the market could plausibly form the basis of a more effective portfolio management strategy. We use neural networks to learn the relevant differential information contained in consecutive financial filings. This approach has several advantages over String edit-distance because it can represent complexities in the difference between documents. Once we vectorize document text into a feature space semantic meaning and hierarchical structure may be learned using the neural net.

The success of this strategy depends on the degree to which the Efficient Market Hypothesis is true. It claims, in weaker and stronger forms, that all relevant information governing the value of securities are already incorporated into the price of the security, which is then the best estimate of the value of that security. Fama et. al. notes that the prices of securities will also over-adjust to new intrinsic values as often as they under-adjust, and may adjust prior to new information being made public or after. This would make any investment strategy based on identifying mispricing nearly-impossible, thus invalidating the existence of abnormal returns over the long run.

If Efficient Markets is true, then no amount of abstraction and parsing can consistently predict outperformance. This complicates our model evaluation, since poor performance may either indicate a poor model or that the task is intrinsically impossible; however, if markets are not efficient and the ”Lazy Prices” results are reproducible over our data set, then we should be able to achieve good results given good models.

2 Data Model

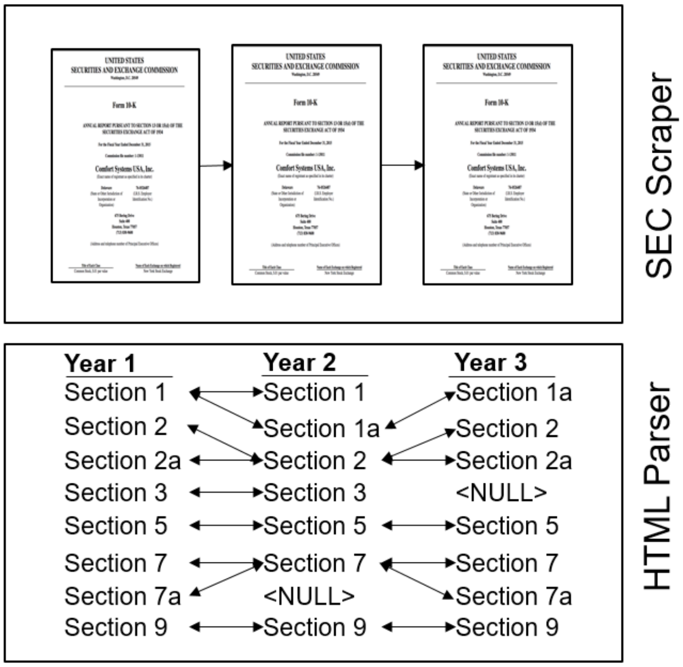


Figure 1: Graphical depiction of scraping and parsing. Parsing is simplified. We created a decision to decide for matching sections when missing subsections or sections are encountered.

2.1 Data Acquisition

We used the SEC Edgar database as our main source for filing information. As of 1994, the SEC has mandated that all companies submit a digital filing of their 10-K forms. These are available in multiple formats - HTML, text, and XBRL (eXtensible Business Reporting Language). Since we are most interested in the textual information, rather than specific descriptions and reports, we focused on acquiring HTML and text documents. EDGAR does not have an API with which we can rest documents. Consequently, downloading the filing documents from SEC Edgar required the development of a scraping tool. Our scraper was loosely based off of the Edgar Python library, but we eventually developed our own expanded scraper to better suit our required functionality.

Our final scraper consumes a list of company stock tickers, and requests listings of filing indices from SEC Edgar. We ingested all available 10-K annual filing documents from all companies listed on the NYSE and NASDAQ exchanges from 1994 to present. The crawler parses the listing year, and identifies the relevant documents to download amongst other attached files, documents, exhibits, etc. It prioritizes HTML documents over text documents to improve our signal to noise ratio, as we can more easily parse and identify edge cases in the HTML form. We then parse the document to extract its individual sections, saving those as well as the entire document, converted to text. Sections were identified by parsing tables and lists of links within the original HTML file.

However, our HTML parser was not able to identify linked sections in all files, and was not at all able to parse .txt-based filings. As such, we developed a second parser to ingest all of our related .txt documents and perform search-based parsing to identify sections. We applied this tool to our HTML files as well, in order to extract any sections that may have previously been missed. In our handling of the downloaded data, we prioritized sections that were parsed directly from HTML, and used the .txt parsed files to augment our data where necessary.

We had to implement extensive error checking in our parser that checks for which section IDs were derived from the raw data, matching possible concatenation errors for each section. Each permutation of possible concatenations between sections is considered and giving a distinct section ID, so that training examples only compare macroscopically alike sections between consecutive years while allowing for the word by word differences we sought to capture. This error checking step was vital to our preprocessing because comparing concatenated sections containing many smaller sections to a true section would misrepresent training examples as containing many more changes than actually contained in the data. As errors were miscellaneous and ranged widely in type (linking errors, different formatting, HTML quirks, and so on), a substantial amount of time on the pipeline to ensure that the data collected were usable.

2.2 Data Preprocessing

Once we acquired our data set, we needed to identify 'valid' pairs of documents to compare, generate their proper embedded representations, and then prepare them as inputs to our neural network. A pair was considered valid if it contained two documents from consecutive years with a matching section ID. Each document was cleaned of any punctuation, numbering, or uppercase lettering. Each word was tokenized, and vectorized using GLoVe representation trained on the Wikipedia dataset. To represent a document, we took the mean of each word embedding in the document. This allows us to compare two documents with variable lengths.

The labels were created using data from Bloomberg Historical Market Capitalization, and are denoted as a one or a zero. A label of 1 corresponds to a 10K section whose differences from the previous year's analogous section yields a positive change in market capitalization one year later. A label of 0 denotes a negative change.

For the target labels, we simplify the task of predicting out-performance by calculating the year-on-year percentage change in market capitalization for each company, then partitioning the changes into two categories, with 0 being worse performance (less than the median for that year) and 1 being the better performance (larger than median for that year). This abstracts away from predicting stock price alone, since prices may change drastically for reasons entirely unrelated to performance, such as stock splits, reverse stock splits, share repurchasing programs, and so on.

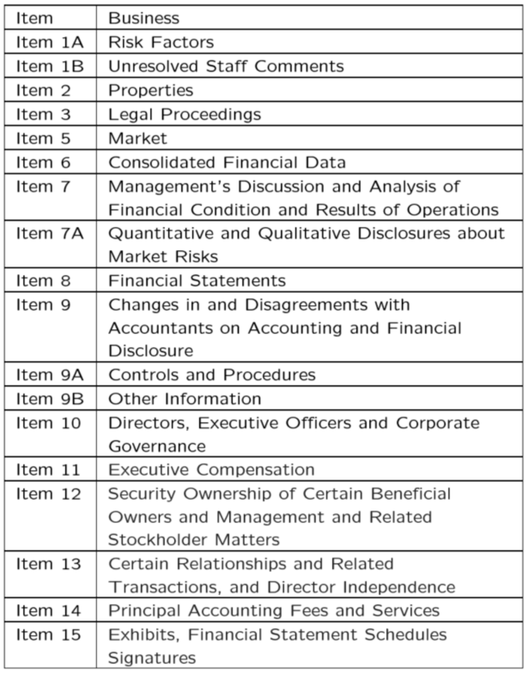
2.3 Outcome Variables

"Lazy Prices" evaluated their similarity measure using a portfolio management application. The portfolio bought companies that exhibited high rates of conversation across financial disclosures and shorted companies who changed. Their performance metric measured the difference in the portfolio's returns compared with the rate of return of common index funds like the SP500. Using their similarity measure managed portfolio, they reported a net return of 30-60 basis points month over month.

Instead of measuring performance by creating our own stock portfolio manager, we trained a model to predict the change of a company's longer term market value. The decision was made to hold a long view to allow the latent information of a company's 10K filings manifest themselves in the stock prices under the assumption of Lazy Pricing. we use the percentage change in market capitalization over the given date range. This allows us to better generalize to differently sized companies, operating under the assumption that similar changes in semantic meaning in paragraphs result in proportional changes to market capitalization. Comparing by year also enables our model to account for overall shifts in the stock market as a whole, such as the dot-com boom, subsequent bust, and the 2008 recession, and consider a company's performance relative to other companies in that year.

We chose two labelling methods for our output variable. The first labelling strategy classified training examples as either outperforming the median change of market capitalization across all US securities for that year or under-performing their expectation. Our alternative labelling strategy classified training examples as a binary label depending on if their market value change was negative or positive year over year. This metric departs from Cohen's usage of stock portfolios so that we are able to capture more information about a company's value. Additionally, our data becomes invariant to the effects of stock splits. We collected approximately 140,000 raw data points from historical market capitalization records using a Bloomberg terminal for each of the filing dates for each company in our data set, and adjust for inflation to prevent naturally rising market capitalization even if firm value hasn't changed.

2.4 Data Representation



Item	Business
Item 1A	Risk Factors
Item 1B	Unresolved Staff Comments
Item 2	Properties
Item 3	Legal Proceedings
Item 5	Market
Item 6	Consolidated Financial Data
Item 7	Management's Discussion and Analysis of Financial Condition and Results of Operations
Item 7A	Quantitative and Qualitative Disclosures about Market Risks
Item 8	Financial Statements
Item 9	Changes in and Disagreements with Accountants on Accounting and Financial Disclosure
Item 9A	Controls and Procedures
Item 9B	Other Information
Item 10	Directors, Executive Officers and Corporate Governance
Item 11	Executive Compensation
Item 12	Security Ownership of Certain Beneficial Owners and Management and Related Stockholder Matters
Item 13	Certain Relationships and Related Transactions, and Director Independence
Item 14	Principal Accounting Fees and Services
Item 15	Exhibits, Financial Statement Schedules Signatures

Figure 2: List of SEC Form 10-K Sections

The Form 10-K is an annual filing that comprehensively describes a company's performance for a fiscal year. All US domestic companies are mandated by the Securities and Exchange Commission (SEC) to file a 10-K each fiscal year.

Our data set contains 9.6 billion tokens extracted from these 10-K filings with a vocabulary size capped at 60,000 over 60,000 documents in total, at more than *90GBs of raw text and html files*.

We started by training word2Vec and document2Vec embeddings on a smaller corpus, as scraping and data cleaning were time-consuming. This approach was bottlenecked by the data collection process. We therefore use an alternative approach through GLoVe vectors over the Wikipedia corpus. Documents represented through this scheme is the arithmetic average of the GLoVe vector for each of the words in that document. We repeat this for each of the 50, 100, and 300 dimensions available.

These representations were developed both at the document and the section level. Our decision to split the data in this fashion was driven by the hypothesis given in the ”Lazy Prices” paper. Analyzing changes at the granularity of each section would result in a more meaningful representation of the document semantics. Since sections many do not experience significant changes year-on-year, a GLoVe-averaged document level representation may have a very low signal-to-noise ratio. Comparing individual sections instead allow us to focus on the semantic differences between smaller components.

Additionally, the ”Lazy Prices” paper considered the relative changes between individual sections, specifically noting that some sections, such as “Item 7. Management’s Discussion and Analysis of Financial Condition and Results of Operations”, were on average much more dynamic than others. This section-based representation also allows us to train models only on an individual section. Since many sections do not experience significant changes year-on-year, a even small change in a normally stagnant section could indicate a larger shift in the company’s material performance.

3 Network Architecture, Hyperparameter Tuning, and Results

To best capture the effect that changes in financial disclosures on performance, we try two different ways to model this change: first as a single input representing the difference in the embedding space between two documents; second as two inputs, each individual representations in the embedding space. We leave the task of distinguish relevant shifts to the network. We also use different dimensions of embeddings: our hypothesis was that more complex spaces may between capture changes; however, in experimentation we found that higher dimensional representations did not increase performance on the validation set. This is likely due to the blowup in number of the parameters and thus the amount of data needed to fully train our models. Again, given the limitation of our corpus by the number of financial disclosures available on EDGAR, we were thus restricted to the 50 dimensional representations.

Our best performing network architecture consisted of two fully-connected hidden layers, each with ReLU activations and L2 regularization. Our objective function was categorical cross-entropy and we use the ADAM optimizer throughout with varying learning rates and decay rates. We avoid deeper networks due to rapidly increasing number of parameters in such networks. We also initially

To further address overfitting, we experiment with dropout layers at different dropout rates between the hidden layers and prior to output. We experimented with different activations for hidden layer, but found that ReLU works best on the validation set.

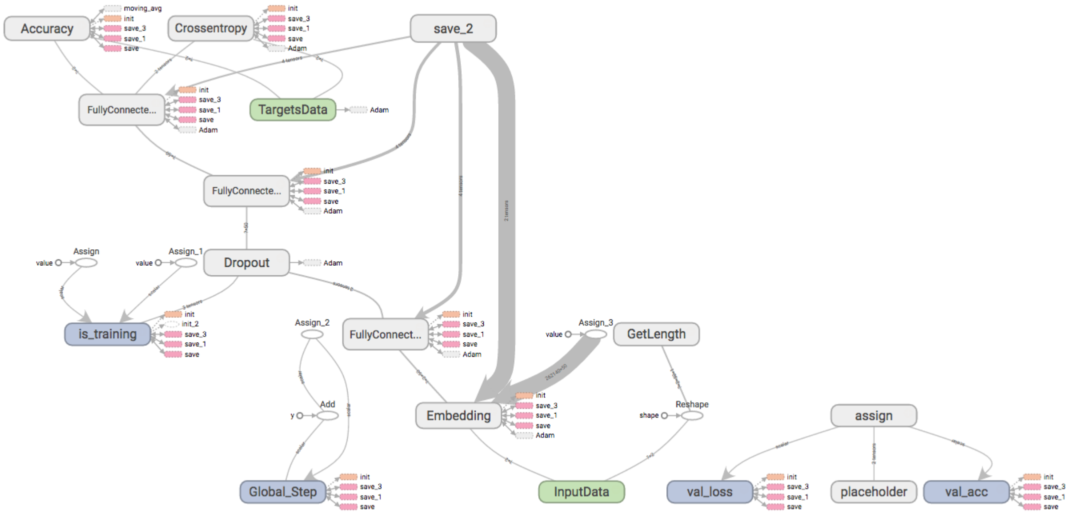


Figure 3: Computational Flow graph for Tensorflow. Tensorflow’s model of our specified feedforward network architecture with two hidden layer, an input layer into a dense embedding layer, and the final output.

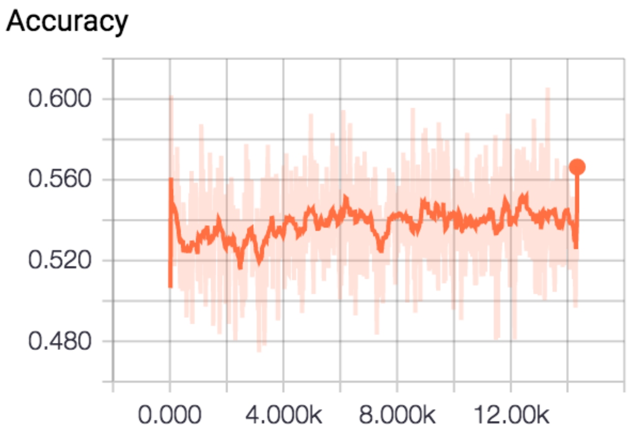


Figure 4: Accuracy over training and validation set for best performing model. Accuracy over validation set is in dark orange. Background is training set (with minibatches).

Input Data	Input Dim.	Parameters	Train Acc.	Test Acc.	Val Acc.
X: 10-K, Y: Median Shift	2x50	20H x 30H x 2Y	0.5687	0.4914	0.4955
10-K, Median Shift	2x50	50I x 2Y	0.6189	0.4979	0.4983
Section 7, Median Shift	2x50	50H x 50H	0.7514	0.493	0.5112
10-K, Median Shift	2x50	50H x 2Y	0.8853	0.5022	0.5023
10-K, +/-	2x50	50 x 50Y x DO x 2Y	0.5956	0.5307	0.5351
Section 7, +/-	2x50	50H x 50H x 2Y	0.8206	**0.5556**	0.5594
Section 1, +/-	2x50	50H x 50H x 2Y	0.6968	0.5172	0.5382
10-K, +/-	2x50	10H x 10H x 2Y	0.513	0.5364	0.548
10-K, +/-	2x50	200H x 2Y	0.5439	0.5372	0.537
10-K, +/-	2x50	5H x 5H x DO x 5H x 2Y	0.5787	0.5407	0.5388
10-K, +/-	1x50	5H x 5H x DO x 5H x 2Y	0.551	0.5371	0.5445

Table 1: Reported test, train, and validation accuracy for a selection of feed-forward neural networks. Accuracy refers to number of exact matches on binary prediction.

4 Discussion

We see predictive ability of our best model on the test set above random. This is an encouraging sign. Given the number of traders and arbitrageurs who seeks to exploit informational inefficiencies in the market, we may have reasonably expected that no model could have picked up the signal hidden amongst the noise. This is the Efficient Markets phenomenon described by Fama.

We also found certain sections performed better than others when predicting if a market cap change would be positive or negative. This corresponds with intuition and "Lazy Prices" findings. Changes year on year are localized to specific sections and items. Hence, where the changes are most exaggerated are where our model managed to distinguish between negative and positive changes. This is especially true for the Management Discussion and Risks sections.

Our choice of document representations isn't ideal - we don't see that performance drastically improves with larger representations at the densely connected layer. This may be due to the choice of granularity we have chosen for the comparisons across 10Ks — differences are captured as well through a smaller dense layer as it is through a larger one. As well, the averaged GloVe vector is a naive approach, and loses the semantic meaning latent in the sequence of words as well as sequence of sub-paragraphs within sections. The possibility exists that averaged GloVe vectors will drown out new information and added sections due to the large fraction of text that consist of legalese boilerplate repeated year on year without change. To realize meaningful changes between years then, we may be looking at a substantially larger requirement on input data larger than the corpus of documents available from the SEC.

5 Next Steps

For concrete next steps, we would like to train word2Vec and Doc2Vec embeddings on the larger corpus we have built up over the course of this project. When initially attempted, we were looking at a corpus across 500 companies and slightly less than 10,000 documents with 328 Million tokens in total. That number has expanded substantially since, to over 6,000 companies with approximately 60,000 filings that can each be split into ten sections on average. This leaves us with over 600,000 sections, and approximately 9.6 billion tokens. This makes training purpose-specific word vectors possible, which may better capture the language and semantics of financial filings. The well-written nature of most 10Ks make them especially amenable to word2Vec training after cleaning.

Our current model assumes that the latent data within each filing will be represented in the market capitalizations a year in the future. In order to more thoroughly explore this and other estimates, we would aggregate market cap data for dates from 1 to 18 months after each 10K's filing date, as well as the mean market cap for the following year. For monthly dates, we take the mean market cap in a window of five days to normalize for daily price fluctuation. With this data, we will be able to vary our estimate for the elapsed time until which the latent data in the filings is materially reflected.

Furthermore, other financial disclosures, such as 8Ks, exist and are available that represent materially relevant information for financial performance over the year that must be disclosed as soon as is known to management. These type of documents are more difficult to integrate into our models. Earlier concerns with insufficient data maybe alleviate by integrating 8Ks.

We would also like to explore more sophisticated models using recurrent neural networks over entire sections — this approach may better preserve the meaning of documents. For example, an attention mechanism may help us naturally hone in on the parts of the documents that change year on year or that which has significant impact as related to market performance.

Finally, aggregating all English financial disclosures for all companies listed in U.S. based exchanges will expand the corpus. These documents are different in structure to 10Ks due to legislation but still constitute a reasonable source of data. T This expansions enables us to bypass some of the model restrictions due to insufficient data described previously. Deeper models and higher-dimension embeddings may meaningfully improve results.

Acknowledgements

We would like to thank the CS224n TAs for assisting with the challenges of this project, Richard Socher for contributing to the model design, and Chris Manning for providing the infrastructure without which this project would not have been possible.

References

[1] Cohen, Lauren and Malloy, Christopher J. and Nguyen, Quoc H. (2016) *Lazy Prices* Available at SSRN: https://ssrn.com/abstract=1658471

[2] Eugene F. Fama. (1965) *http://download.tensorflow.org/paper/whitepaper2015.pdf* Journal of Business. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.365.9468

[2] Mikolov, Tomas. Ilya Sutskever. Kai Chen. Greg Corrado. Jeffrey Dean. (1995) *Distributed Representations of Words and Phrases and their Compositionality* Available at: https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[3] Andrew M. Dai. Christopher Olah. Quoc V. Le. (2015) *Document Embedding with Paragraph Vectors* Available at: https://arxiv.org/pdf/1507.07998.pdf

[4] Jeffrey Pennington. Richard Socher. Christopher D. Manning. (2014) *GloVe: Global Vectors for Word Representation* Available at: https://nlp.stanford.edu/pubs/glove.pdf

[5] Martin Abadi et al. (2015) *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems* Available at: http://download.tensorflow.org/paper/whitepaper2015.pdf

[6] Jiang Wang. (2016) *Machine Comprehension Using Match-LSTM and Answer Pointer*

[7] Hagiwara, M, Ke, Y. (2016) *Alleviating Overfitting for Polysemous Words for Word Representation Estimation Using Lexicons* CoRR, abs/1612.00584.

[8] Figueroa, R. L., Zeng-Treitler , Q., Kandula, S., Ngo, L. H. (2012) *Predicting sample size required for classification performance* BMC Medical Informatics and Decision Making, 12, 8. http://doi.org/10.1186/1472-6947-12-8

[9] Mikolov, Tomas, Yih, Scott Wen-tau, and Zweig, Geoffrey. *Linguistic regularities in continuous space word representations* In NAACL HLT, 2013d.

[10] Turney, Peter D. and Pantel, Patrick. *From frequency to meaning: Vector space models of semantics*. Journal of Artificial Intelligence Research, 2010.

[11] Rumelhart, David E, Hinton, Geoffrey E, and Williams, Ronald J. *Learning representations by back-propagating errors* Nature, 323(6088):533–536, 1986.

[12] Socher, Richard, Lin, Cliff C, Ng, Andrew, and Manning, Chris. *Parsing natural scenes and natural language with recursive neural networks* In Proceedings of the 28th International Conference on Machine Learning (ICML- 11), pp. 129–136, 2011b.

[13] Smith, Stephen. *The Road to TensorFlow – More on Optimization*. (2016, October 04). Retrieved March 22, 2017, from https://smist08.wordpress.com/2016/10/04/the-road-to-tensorflow-part-10-more-on-optimization/

[14] Michael Auli, Michel Galley, Chris Quirk, and Geoffrey Zweig. Joint language and translation modeling with recurrent neural networks In EMNLP, pages 1044–1054, 2013.

[15] Yoav Goldberg and Omer Levy. *word2vec explained: deriving Mikolov et al. negative-sampling word-embedding method*. arXiv preprint arXiv:1402.3722, 2014.

[16] Alex Graves. *Supervised sequence labelling with recurrent neural networks* volume 385. Springer, 2012.