Numeracy of Language Models: Joint Modelling of Words and Numbers

Georgios P. Spithourakis

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy
of
University College London.

Department of Computer Science
University College London

May 27, 2019

I, Georgios P. Spithourakis, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Numeracy and literacy are the abilities to understand and work with numbers and words, respectively. While both skills are necessary for reading and writing documents in clinical, scientific, and other technical domains, existing statistical language models focus on words to the expense of numbers: numbers are ignored, masked, or treated similarly to words, which can obscure numerical content and cause sparsity issues, e.g. high out-of-vocabulary rates. In this thesis, we investigate whether the performance of neural language models can be improved by i) considering numerical information as additional inputs and ii) explicitly modelling the output of numerical tokens.

In experiments with numbers as input, we find that numerical input features improve perplexity by 33% on a clinical dataset. In assisted text entry and verification tasks, numerical input features improve recall from 25.03% to 71.28% for word prediction with a list of 5 suggestions, keystroke savings from 34.35% to 44.81% for word completion, and F1 metric by 5 points for semantic error correction. Numerical information from an accompanying knowledge base helps improve performance further.

In experiments with numerical tokens as output, we consider different strategies, e.g. memorisation and digit-by-digit composition, and propose a novel neural component based on Gaussian mixture density estimation. We propose the use of regression metrics to evaluate numerical accuracy and an adjusted perplexity metric that accounts for the high out-of-vocabulary rate of numerals. Our evaluation on clinical and scientific datasets shows that perplexity can be improved by more than 2 and 4 orders of magnitude, respectively, by modelling words and numerals with

different sub-models through a hierarchical softmax. For the same datasets, our proposed mixture of Gaussians model achieved a 32% and 54% reduction of mean average percentage errors over the contender strategy, digit-by-digit composition. We conclude with a critical reflection of this thesis and suggestions for future work.

Impact Statement

This work investigates the relationships between words, numbers, numerals, and several models, particularly language models, in statistical tasks and applications in the area of natural language processing.

The work contributes to highlight the strong connections between words and numbers and exemplifies how integration of word and number inputs can lead to improvements in tasks and downstream applications of language models. Specifically, our insights could improve the quality of service for assisted text entry systems (word prediction and completion) and for assisted number-based proof-reading systems (semantic error detection and correction). Application of such systems in clinical domains could facilitate the composition of clinical reports and reduce the amount of errors that could adversely affect the quality of heath-care.

Furthermore, this work contributes to the methodology of neural language modelling by proposing models with alternative strategies (hierarchical, composition, from continuous density, and combination of strategies) for modelling numerals and for predicting numbers; our evaluations reveal that the proposed models can improve the (adjusted) perplexity evaluation metric by 100 and 10,000 times in a clinical and scientific domain, respectively. Applications of language models in other number-intensive domains could benefit from these improvements. The methodological contributions in evaluations of language models (e.g. only on numerals), as well as the insights for the nature of numerals as special words that confer numerical information, can be applied to guide and promote future research.

This stream of work has already generated publications in leading conferences, such as EMNLP and ACL.

Acknowledgements

I submit my gratitude to Professor Sebastian Riedel; his supervision made this work possible. For everything NLP and everything non-NLP that you have taught me, thank you Sebastian!

All credit for the clinical aspects of this work is to Professor Steffen E. Petersen, who was my second supervisor. Thank you Steffen!

Thank you Isabelle Augenstein, Pontus Stenetorp, Andreas Vlachos, Jeff Mitchell, Pasquale Minervini, Guillaume Bouchard, and Jason Naradowsky, for your guidance during our shared time at University College London's Machine Reading group. Thank you Marzieh Saeidi, Matko Bosnjak, Tim Rocktäschel, Ivan Sanchez, and Johannes Welbl, for going through study, work, and fun times together.

Many thanks to Arpit Mittal and to Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan, with whom I had the pleasure to work during my internships at Amazon and Microsoft Research. Many thanks to my examiners, Spiros Denaxas and Ivan Titov, that engaged me with their discussion and constructive feedback. I thank the people at the Farr Institute of Health Informatics Research, where the funding for this work came from, and the people at University College London, where the rest of the resources and support came from. Thank you all!

Thank you $\mu\alpha\mu\acute{\alpha}$, $\mu\pi\alpha\mu\pi\acute{\alpha}$, $M\alpha\rho\iota\acute{\alpha}\nu\nu\alpha$.

Contents

1 Introduction **19**

1.1 Motivation 19

1.2 Research Questions 21

1.3 Contributions 23

1.4 List of Publications 24

1.5 Structure of the Thesis 24

2 Background **27**

2.1 Categorical Variables and Related Tasks 28

2.1.1 Classification 28

2.1.2 Language Modelling 33

2.1.3 Text Prediction 37

2.1.3.1 Word Prediction 37

2.1.3.2 Word Completion 38

2.1.4 Error Detection and Correction 39

2.1.4.1 Error Detection 40

2.1.4.2 Error Correction 40

2.2 Numerical Variables and Related Tasks 41

2.2.1 Regression 42

2.3 Words and Numbers 43

3 Related Work **47**

3.1 Language Modelling 47

3.2 Text Prediction 51
3.3 Error Detection and Correction 53
3.4 Numbers in Natural Language Processing 56
3.5 Applications of Natural Language Processing 59

4 Establishing the Connection between Words and Numbers 61
4.1 Classification: From Numbers to Words 65
4.1.1 Approach 66
4.1.2 Data 66
4.1.3 Experimental Setup 70
4.1.4 Results 71
4.1.5 Discussion 74
4.2 Regression: From Words to Numbers 77
4.2.1 Approach 77
4.2.2 Data 78
4.2.3 Experimental Setup 78
4.2.4 Results 82
4.2.5 Discussion 83

5 The Effect of Numbers on Language Modelling 85
5.1 Language Modelling 86
5.1.1 Approach 87
5.1.2 Data 88
5.1.3 Experimental Setup 90
5.1.4 Results 91
5.1.5 Discussion 93
5.2 Application: Text Prediction 94
5.2.1 Approach 96
5.2.2 Data 97
5.2.3 Experimental Setup 97
5.2.4 Results 98

5.2.5 Discussion . . . 101
5.3 Application: Semantic Error Detection and Correction . . . 102
5.3.1 Approach . . . 104
5.3.2 Data . . . 105
5.3.3 Experimental Setup . . . 106
5.3.4 Results . . . 107
5.3.5 Discussion . . . 109

6 Modelling Numerals and Predicting Numbers with Language Models 111
6.1 Modelling Numerals with Language Models . . . 112
6.1.1 Approach . . . 112
6.1.2 Data . . . 117
6.1.3 Experimental Setup . . . 118
6.1.4 Results . . . 119
6.1.5 Discussion . . . 124
6.2 Predicting Numbers with Language Models . . . 127
6.2.1 Approach . . . 127
6.2.2 Data . . . 128
6.2.3 Experimental Setup . . . 128
6.2.4 Results . . . 128
6.2.5 Discussion . . . 129

7 Conclusion 133
7.1 Critical Reflection . . . 135
7.2 Future Work . . . 137

Appendices 139

A Clinical Dataset 139

Bibliography 142

List of Figures

1.1 Basic concepts 20

3.1 Our position among approaches to Language Modelling. 51
3.2 Our position among approaches to Text Prediction. 53
3.3 Our position among approaches to Error Detection and Correction. . 56

4.1 Cardiac cycle. 63
4.2 Confusion matrix for the *text*∪*kb* model 73
4.3 Decision areas of the *kb* model 76
4.4 Histograms of EDV for class of dilation. 79
4.5 Histograms of EF for class of dilation. 80

5.1 Ratios of word probabilities under different language models 92
5.2 Semantic error correction using language models. 105
5.3 Sensitivity of semantic error correction to numbers 108

6.1 Challenges for modelling numerals with language models 113
6.2 Mixture of Gaussians model. 117
6.3 Cosine similarities of embeddings for numerals. 122
6.4 Cosine similarities of embeddings for digits. 123
6.5 Distributions of significant digits. 124
6.6 Example model predictions. 130

List of Tables

1.1 Overview of models in this thesis with examples for inputs and outputs 22
1.2 Research questions 22

2.1 Layout of the confusion matrix for binary classification 31

4.1 Example of data from the clinical domain 64
4.2 Common text patterns for LV dilation 67
4.3 Statistics of document annotation 67
4.4 Statistics of attributes missing from the KB 68
4.5 Statistics of EDV and EF attributes 70
4.6 Results for word prediction 71
4.7 Breakdown of results for word prediction 72
4.8 Most important features from the *text* model 74
4.9 Most important features from the *text*∪*kb* model 75
4.10 Results for regression for EF and EDV 81
4.11 Breakdown of results for regression for EF and EDV 82
4.12 Most important features from the *text* model 83

5.1 Example of text with dependencies on a KB and on numbers 86
5.2 Example of KB lexicalisation 89
5.3 Statistics for clinical dataset 90
5.4 Results for language modelling 91
5.5 Task examples for word prediction and completion 95
5.6 Results for word prediction 98
5.7 Results for word completion 98

5.8 Word prediction for sample document 100
5.9 Task examples for semantic error detection and correction 103
5.10 Confusion sets 106
5.11 Results for semantic error detection 107
5.12 Results for semantic error correction 108

6.1 Statistical description of the clinical and scientific datasets 118
6.2 Results for language modelling (adjusted) 120
6.3 Results for language modelling (unadjusted) 120
6.4 Examples of numerals for each strategy of the combination model . 125
6.5 Results for regression 129

A.1 Statistics for patients in clinical dataset 140
A.2 Number of CMRs requested, analysed, or finalised per person . . . 141

Chapter 1

Introduction

'The numbers are the key to everything'
— Nicolas Cage as 'John Koestler',
in *The Knowing*

Literacy and *numeracy* refer to the ability to comprehend, use, and attach meaning to words and numbers, respectively. *Words* are *categorical* labels devised by humans to categorise, document, and communicate information, while *numbers* are abstract mathematical objects that facilitate *numerical* reasoning. *Numerals* are closely related to both words and numbers: a numeral is a word that refers to a number; thus, the ability to understand numerals is essential for numeracy, but it can also be important to literacy. Figure 1.1 shows the relations among these concepts.

This thesis is about the joint modelling of data of different types: words, numbers, and numerals.[1] We seek to improve the performance of models by i) incorporating inputs of different types, e.g. numerical inputs to models for word prediction, and ii) by proposing mechanisms for modelling outputs of different types, e.g. numerical outputs from word prediction models. Our emphasis is on language models, which are probabilistic models of texts, that is, sequences of words and numerals.

1.1 Motivation

Numbers and Words Numbers are abstract mathematical objects that can be used to measure, count, and label; this contrasts with words, which can only act as labels.

[1] We draw a clear distinction between the terms 'numbers' and 'numerals', and our usage of the term 'words' excludes numerals.

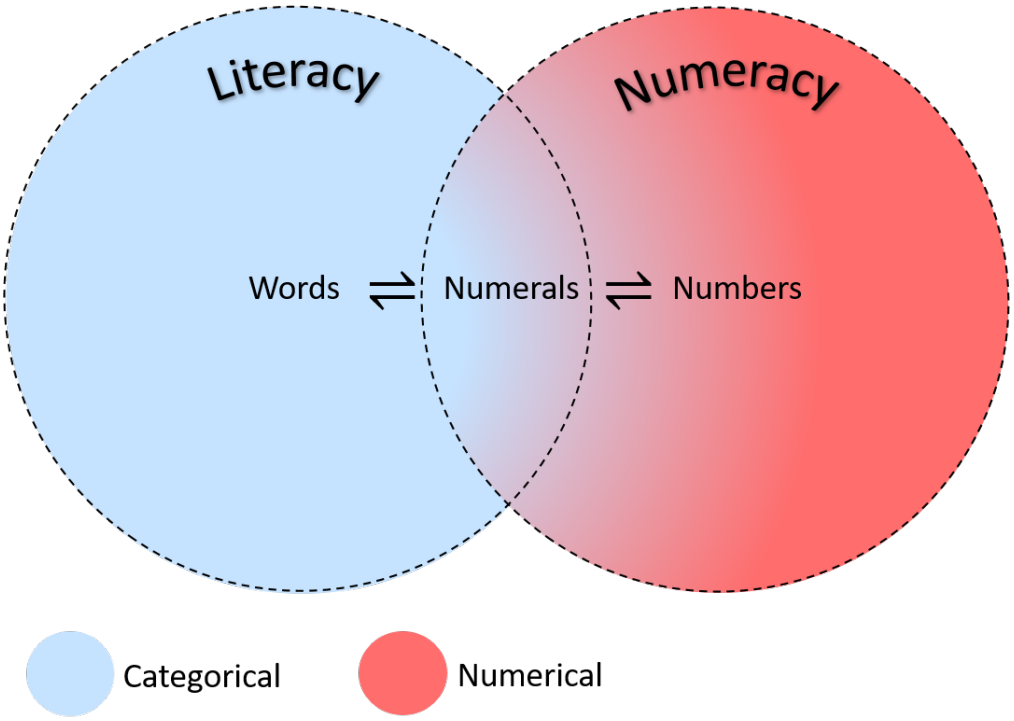

Figure 1.1: Basic concepts. Literacy and numeracy are the abilities to work with words and numbers, respectively. Words are categorical labels, and numbers are abstract numerical objects. Numerals are categorical labels (like words) for numbers.

Numbers have the continuous property of *numerical magnitude*, and they can be ordered according to it. This continuity facilitates generalisation to numbers that have never been seen before. For example, if someone describes two persons with heights 1.8 metres and 2.1 metres as 'tall', we would expect them to use the same word to describe anyone with a height between those values. On the contrary, generalisation across words is harder, or even impossible, because they are categorical, i.e. they have no magnitude and no natural ordering. For example, if someone describes two persons called 'Sarah' and 'John' as 'tall', we would have no information on how they would describe people with different names. Moreover, relevant information can be better represented as numbers (e.g. height) or as words (e.g. biological sex: 'male' or 'female'). Therefore, having both numbers and words as inputs to models can help capture more information and improve generalisation to unseen data.

Numerals and Words Numerals are symbols that refer to numbers, just as words are symbols that refer to other objects and ideas. Unlike most words, new numerals can be procedurally generated to refer to never before encountered numbers (and

there are infinite of them), i.e. numerals come from an open vocabulary. In many applications of natural language processing, numerals are often neglected and low-resourced, e.g. they are often simply filtered from text data or masked (Mitchell and Lapata, 2009), and there are only $15,164$ (3.79%) numerals in the vocabulary of GloVe's $400,000$ pretrained embeddings (Pennington et al., 2014). In language modelling, i.e. statistical estimation of probabilities of texts, numerals are often treated like all other words: most language models assume that their inputs and outputs are categorical, i.e. they have no magnitude. However, the number that underlies the numeral could be retrievable, up to an extent, e.g. the numeral in the text 'with a height of *2* metres, he is a tall' is likely to represent number 2 or another number with a similar magnitude (e.g. 1.99, 2.001, etc.). The number could then be used as a continuous numerical input to the model to help the model predict certain words, e.g. 'tall', and generalise over unseen numerals. Finally, the modelling of numerals could be improved by using an alternative output mechanism that accounts for the smoothness of the underlying number.

Language Models The emphasis of this thesis is on language models (LMs), which are statistical models that assign a probability over texts (sequences of words and numerals). Language models can often help with other tasks, such as speech recognition (Mikolov et al., 2010; Prabhavalkar et al., 2017), machine translation (Luong et al., 2015; Gülçehre et al., 2017), text summarisation (Filippova et al., 2015; Gambhir and Gupta, 2017), question answering (Wang et al., 2017), semantic error detection (Rei and Yannakoudakis, 2017), and fact checking (Rashkin et al., 2017). Improving the performance of language models may lead to improvements in downstream applications. Therefore, we seek to bring about these improvements by providing language models with mechanisms for the input and output of numerals and numbers.

1.2 Research Questions

This thesis is about the joint modelling of data of different types: words, numbers, or numerals. We seek to improve the performance of models by i) incorporating

Table 1.1: Overview of models in this thesis with examples for inputs and outputs. The text type is a sequence of words and numerals.

Type			Example			Approach	
input	→	**output**	**input**	→	**output**	**model**	**chapter**
numbers	→	words	*height*=2.1	→	*class*='tall'	classifier	Ch. 4
numbers	→	texts	*height*=2.1	→	'John is tall'	LM	Ch. 5
words	→	numbers	'tall'	→	*height*=2.1	regressor	Ch. 4
words	→	numbers	'John's height is [...]'	→	*value*=2.1	LM	Ch. 6
numerals	→	words	'John's height is 2.1m'	→	*class*='tall'	classifier	Ch. 4
numerals	→	texts	'John's height is 2.1m'	→	'John is tall'	LM	Ch. 5
words	→	numerals	'John's height is [...]'	→	'[...] 2.1 [...]'	LM	Ch. 6

Table 1.2: Research questions.

Research Questions		
Q.1	Can inputting numbers into a model that predicts words (classifier) improve its performance?	Ch. 4
Q.2	Can inputting words into a model that predicts numbers (regressor) improve its performance?	Ch. 4
Q.3	Can inputting numbers into a language model improve its ability to model/predict/correct texts (sequences of words and numerals)?	Ch. 5
Q.4	How can we improve the ability of language models to model numerals?	Ch. 6
Q.5	How can we improve the ability of language models to predict numbers?	Ch. 6

inputs of different types, e.g. numerical inputs to models for word prediction, and ii) by proposing mechanisms for modelling outputs of different types, e.g. numerical outputs from word prediction models. Table 1.1 shows an overview of models with various possible types for inputs and outputs found in the thesis.

Table 1.2 lists the research questions that we address in this thesis. Question Q.1 is a preliminary question that investigates the importance of numbers in models that predict words, i.e. classifiers; likewise, question Q.2 is a preliminary question that investigates the importance of words in models that predict numbers, i.e. regressors. Question Q.3 is a preliminary question that extends question Q.1 to language models for language modelling and downstream applications (text prediction, semantic error detection and correction). Questions Q.4 and Q.5 are concerned with finding improved mechanisms for modelling numerals and predicting number with language models.

1.3 Contributions

The main contributions of this thesis are as follows:

1. We find a strong connection between words and numbers, and we achieve improvements in several tasks (classification, regression, language modelling, word prediction and completion, error correction and detection) by using inputs that contain both words and numbers.

2. We propose two mechanisms (magnitude-dependent embeddings and conditioning on lexicalised KB) for incorporating numerical inputs into language models. These lead to improvements for language modelling and several downstream applications of language models (word prediction and completion, error correction and detection). The improvements from the two mechanisms are mostly orthogonal and incremental.

3. We find that language models perform much worse on numbers than on other words, possibly because of the high OOV-rate for numerals. This might lead to significant failures of existing models in number-based applications. This behaviour is not immediately obvious from standard evaluations of language models (unadjusted perplexity evaluated on all tokens).

4. We propose several alternative strategies (hierarchical, compositional, from continuous probability density, and combination of strategies) for modelling

numerals and predicting numbers with language models. We find that for modelling the best perplexity is achieved with a combination of strategies, and for prediction with the continuous density strategy. There are indications that a different strategy can be suitable for different contexts.

1.4 List of Publications

The work in this thesis is partly based on the following publications by the author (in **bold**):

- **G. P. Spithourakis**, S. Riedel. Numeracy for Language Models: Evaluating and Improving their Ability to Predict Numbers. *ACL 2018*.
- **G. P. Spithourakis**, I. Augenstein, S. Riedel. Numerically Grounded Language Models for Semantic Error Correction. *EMNLP, 2016*.
- **G. P. Spithourakis**, S. E. Petersen, S. Riedel. Clinical Text Prediction with Numerically Grounded Conditional Language Models. *LOUHI - 7th International Workshop on Health Text Mining and Information Analysis, EMNLP 2016*.
- **G. P. Spithourakis**, S. E. Petersen, and S. Riedel. Harnessing the predictive power of clinical narrative to resolve inconsistencies and omissions in EHRs. *2nd Workshop on Machine Learning for Clinical Data Analysis, Healthcare and Genomics, NIPS 2014*.

1.5 Structure of the Thesis

In this Chapter, we introduced the concepts of words, numbers, and numerals, and we discussed the relations between them and with literacy, numeracy, and to categorical and numerical data. We motivated and formulated the research questions for the remainder of the thesis (Table 1.2), and we presented the contributions of this work and list of publications by the author. The rest of the thesis is structured as follows:

- In Chapter 2, we introduce the concepts of categorical and numerical variables, and we present models and evaluations for the tasks of classification, language modelling, text prediction (word prediction and completion), error detection and correction and regression.

- In Chapter 3, we position this work within the broader literature by discussing related work on the topics of language modelling, text prediction, error detection and correction, and numbers in natural language processing.

- In Chapter 4, we address research questions Q.1 and Q.2, by investigating a classification and a regression task, with numbers and words as inputs, respectively.

- In Chapter 5, we address research question Q.3, by investigating language models with numbers as inputs in the tasks of language modelling, text prediction (word prediction and completion), and semantic error detection and correction.

- In Chapter 6, we address research questions Q.4 and Q.5, by investigating language models with different strategies for modelling numerals and for predicting numbers, respectively.

- In Chapter 7, we conclude with an overall discussion that critically reflects on this work and proposes pathways to future research.

An overview of models and typical task signatures in each chapter can be found in Table 1.1.

Chapter 2

Background

'I've always believed in numbers and the equations and logics that lead to reason ... It is only in the mysterious equations of love that any logic or reasons can be found'

— John F. Nash

Many problems, including most problems in this thesis, involve finding relationships between variables, in particular how input variables (independent variables) affect an output variable (dependent variable). A variable is any quantity or quality of interest, e.g. a person's age, sex, year and country of birth, height, weight, eye and hair colour, etc, that takes different values in different situations, e.g. among different persons. There are two main types of variables:

- **Categorical Variables** These variables are associated with qualitative attributes, e.g. country of birth, eye and hair colour, etc. The values of categorical variables are categorical labels, i.e. words.
- **Numerical Variables** These variables are associated with quantitative attributes, e.g. age, height, weight, etc. The values of numerical variables are numbers.

Supervised Learning Problem Let x_i be the input variables or input features and let y_i be an output or target variable that we want to predict. Let $\mathscr{X}$ and $\mathscr{Y}$ be the space of input and output values, respectively. Let $D = \{(x_i, y_i) | i = 1, ..., N\}$ be a training

set, where each pair (x_i, y_i) is called a training example. The supervised learning problem can then be defined as: given a training set, learn a function $f : \mathscr{X} \to \mathscr{Y}$, so that $f(x)$ is a 'good' predictor of the corresponding value of y. *Training* refers to the process of learning the function from data, and *evaluation* refers to the process of finding how good the predictor is.

Probabilistic Approaches A random variable Y is a variable[1] that depends on a random process and takes a value y from a set, e.g. the random variable 'toss of a coin' can take any value from the set {*'heads', 'tails'*}. Likewise, random variables can be categorical or numerical, depending on the domain that they take their values from.

Structure of Chapter This chapter is structured as follows:

- In Section 2.1, we present models and tasks for categorical variables;
- In Section 2.2, we present models and tasks for numerical variables;
- In Section 2.3, we provide some additional information on the topics of words, numbers, and numerals.

2.1 Categorical Variables and Related Tasks

A *categorical variable* is a variable whose value comes from a limited set of items. For example, a statement can be 'True' or 'False', a patient's condition can be described as 'normal', 'mild', or 'severe', a word can be any word from the vocabulary of a language, etc. For a categorical random variable Y, the probability that it takes on a particular value y is called its *probability mass function (PMF)*, $p(y) = P(Y = y)$, or simply its probability function.

2.1.1 Classification

Task Definition *Classification* is the problem of assigning an object to a category or class based on features of the object, e.g. assign a patient's condition to a level from an intensity scale {*'normal', 'mild', 'moderate', 'severe', 'critical'*} based on

[1] According to a stricter, but less practical for our purposes, definition, a random variable is a function from the sample space of an experiment to the set of real numbers."

the patient's health record. More formally, a classification problem is a supervised learning problem, where the target variable is a categorical variable. The problem is known as *binary classification* when there are only two possible values for the target variable and as *multi-class classification* when there are more than two.

Probabilistic Classification A probabilistic approach to classification assumes that for an instance x its class y is the value of a categorical random variable from a set of possible labels X. The conditional distribution of the class is modelled with a categorical distribution[2] using the softmax function:

$$\begin{aligned} p(y|x;\theta) = \text{softmax}\left(\psi_\theta(y;x)\right) = \tfrac{1}{Z} e^{\psi_\theta(y;x)} \\ Z_\theta = \sum_{y \in V} e^{\psi_\theta(y;x)}, \end{aligned} \tag{2.1}$$

where θ are the parameters of the model, the normaliser Z_θ is known as the partition function, and $\psi_\theta(y;x)$ is a score function that measures the agreement between every class y and the instance x. A commonly used score function is the dot product between input and output feature vectors:

$$\psi_\theta(y;x) = \phi_{x;\theta}(x)^T \phi_{y;\theta}(y), \tag{2.2}$$

where $\phi_{x;\theta}(x) \in \mathbb{R}^d$ and $\phi_{y;\theta}(y) \in \mathbb{R}^d$ are vector feature representations of the input x and output y, respectively. The probabilistic classifier can produce a 'hard' prediction by assigning the instance to the class with the highest probability:

$$\hat{y} = \underset{y' \in V}{\text{argmax}}\, p(y'|x;\theta). \tag{2.3}$$

For linear score functions, this model is called the *log-linear classifier*.

Training Training the classifier refers to selecting the best configuration for its parameters θ on the basis of an observed training set $D = \{(x_i, y_i), i = 1, ..., N\}$ of N pairs of instances x_i and their labels y_i.

[2] In natural language processing, the categorical distribution is often called the 'multinomial distribution'. We break with that tradition for the sake of clarity.

Maximum Likelihood Maximum likelihood estimation is a statistical method for estimating the model's parameters from data. The parameters are selected to maximise the likelihood that the process described by the model produced the data that were actually observed. Equivalently, the negative log likelihood is minimised:

$$\begin{aligned} -\ln L(\theta|D) &= -\ln \prod_{i=1}^{N} p(y_i|x_i;\theta) \\ &= -\sum_{i=1}^{N} \ln p(y_i|x_i;\theta) \\ &= N \ln Z_\theta - \sum_{i=1}^{N} \ln \psi_\theta(y_i, x_i) \end{aligned} \tag{2.4}$$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \left[-\ln L(\theta|D)\right] \tag{2.5}$$

The optimisation problem can be solved using gradient descent techniques.

Minimum Cross-Entropy Another way to derive a parameter estimate is through minimising an information-theoretic training objective. Cross-entropy measures how useful the model distribution p is in order to predict data that come from a true distribution q:

$$\begin{aligned} \mathscr{H}_{y|x}(q,p) &= -\sum_{y \in V} q(y|x) \ln p(y|x) \\ &= -E_q\left[\ln p(y|x)\right] \end{aligned} \tag{2.6}$$

Considering the training data as a sample from the true distribution q, we can derive a practical Monte-Carlo approximation:

$$\mathscr{H}_{y|x}(D,p) = -\frac{1}{N} \sum_{i=1}^{N} \ln p(y_i|x_i) \tag{2.7}$$

This approximation of the cross-entropy is proportional to the negative log-likelihood of the model, and it can minimised at training time to obtain estimates for the parameters of the model.

Evaluation for Binary Classification The performance of a classifier is evaluated by comparing the predicted labels with the observed labels on a held out test set. We are interested in the counts for the various combinations of the target variable

Table 2.1: Layout of the confusion matrix for binary classification.

		actual class	
		yes	no
predicted class	yes	tp (true positive)	fp (false positive)
	no	fn (false negative)	tn (true negative)

and predictions:

- **True Positives (TP)** These are cases in which we predicted yes, and the label is yes;
- **True Negatives (TN)** We predicted no, and the label is no;
- **False Positives (FP)** We predicted yes, but the label was no;[3]
- **False Negatives (FN)** We predicted no, but the label was yes.[4]

The above counts are presented as a contingency table, where each row represents a predicted class, and each column an actual class, called the *confusion matrix*. Table 2.1 shows the typical layout of the confusion matrix. Various other evaluation metrics can be derived from the confusion matrix:

- **Accuracy** This metric measures the overall percentage of predictions that were correct. Accuracy is not informative of model performance per class. In some cases of imbalanced classes, a model with high accuracy can have no useful predictive power for the rare class. Accuracy is calculated as:

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{2.8}$$

- **Precision and Recall** These metrics measure model performance for a specific class of interest. Precision measures the percentage of the instances classified as the interest class that actually belong in that class, i.e. how exact

[3] Also known as Type I error.
[4] Also known as Type II error.

the model is at assigning instances to a class. Recall measures the percentage of instances from the interest class that were classified to that class, i.e. how thorough the model is at identifying instances from a class. In binary classification, we are usually interested in the positive ('yes' in Table 2.1) class, and its precision and recall are calculated as:

$$Precision = \frac{tp}{tp + fp} \tag{2.9}$$

$$Recall = \frac{tp}{tp + fn} \tag{2.10}$$

- **F-score** This metric summarises both precision and recall, using a weight to control the importance of each:

$$F_\beta = (1 + \beta^2)\frac{precision \cdot recall}{\beta^2 * precision + recall}, \tag{2.11}$$

where $\beta \in \mathbb{R}^*$ is a weight that controls the contribution of the precision and recall. For $\beta = 1$, both are equally important, and the resulting F_1 metric is equivalent to their harmonic mean:

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{2.12}$$

Evaluation for Multiclass Classification In the multiclass case, the confusion matrix can be extended with additional rows/columns. Precision, recall, and F-score are defined for each class, which gives the finest level granularity and is useful when high performance in specific classes is desired. Otherwise, averaged versions of these metrics exist that summarise the classifier's performance over all classes. Different types of averaging can be performed on the data:

- **Micro Average** Calculate metrics globally by counting the total true positives, false negatives and false positives.

- **Macro Average** Calculate metrics for each label, and find their unweighted mean, which does not take label imbalance into account (this assumes that all

classes are equally important).

- **Weighted Average** The weighted average of the metric across all classes, where the weights are the class proportions in the data (assumes that class importance correlates with its frequency in the data).

The micro-averaged precision, recall, F1, and accuracy will always have the same value. The weighted-averaged F1 might not coincide with the harmonic mean of the weighted-averaged precision and recall.

2.1.2 Language Modelling

Task Definition Let $s_1, s_2, ..., s_L$ denote a document, where s_t is the token at the t-th position. *Language modelling* is the task of estimating the probability of observing a given document, i.e. $p(s_1, ..., s_L)$. This involves learning a Language Model (LM), which is an estimator of the probability of the next token given its previous tokens, i.e. $p(s_t|s_1, ..., s_{t-1})$. Often, a neural network is used to estimate this probability, and several different neural architectures for language models have been proposed (see *Approaches* in Section 3.1). In this section, we present some technical background for the approach of language modelling with recurrent neural networks. Recurrent neural LMs estimate this probability by feeding embeddings, i.e. vectors that represent each token, into a Recurrent Neural Network (RNN) (Mikolov et al., 2010).

Input Layer This layer calculates representations for the tokens, called *embeddings*. Several different options for defining embeddings are listed bellow:

1. **Token-Based Embeddings** Tokens are most commonly represented by a D-dimensional dense vector that is unique for each word from a vocabulary $\mathcal{V}$ of known words. This vocabulary includes special symbols (e.g. 'UNK') to handle out-of-vocabulary tokens, such as unseen words or numerals. Let w_s be the one-hot representation of token s, i.e. a sparse binary vector with a single element set to 1 for that token's index in the vocabulary, and $E \in \mathbb{R}^{D \times |\mathcal{V}|}$ be the token embeddings matrix. The token embedding for s is the vector:

$$e_s^{\text{token}} = E w_s. \tag{2.13}$$

2. **Character-Based Embeddings** A representation for a token can be build from its constituent characters (Luong and Manning, 2016; Santos and Zadrozny, 2014). Such a representation takes into account the internal structure of tokens. Let $d_1, d_2, ..., d_N$ be the characters of token s. A character-based embedding for s is the final hidden state of a D-dimensional character-level RNN:

$$e_s^{\text{chars}} = \text{RNN}(d_0, d_1, ...d_L). \tag{2.14}$$

3. **Gated Token-Character Embedding** Either of the previous embeddings can be better suited to represent the token. Miyamoto and Cho (2016) combine both using a learnt gating function per hidden unit:

$$\begin{aligned} e_s^{gated} &= g \odot e_s^{token} + (1-g) \odot e_s^{chars} \\ g &= \sigma\left(W e_s^{token} + W' e_s^{chars}\right) \end{aligned} \tag{2.15}$$

where W and W' are transformation matrices with suitable dimensions, and $\sigma(.)$ is the sigmoid function.[5]

Recurrent Layer The computation of the conditional probability of the next token involves recursively feeding the embedding of the current token e_{s_t} and the previous hidden state h_{t-1} into a D-dimensional token-level RNN to obtain the current hidden state h_t.

1. **Vanilla RNN** This is uses the simplest recurrent function:

$$h_t = \tanh(Ah_{t-1} + Bx_t), \tag{2.16}$$

where tanh is the hyperbolic tangent function. Vanilla RNNs often suffer from the vanishing or exploding gradient problems (Pascanu et al., 2013; Bengio et al., 1994): in the exploding gradient problem, the norm of the gradient increases explosively during training, which makes training numerically un-

[5] We follow the trend of referring to a particular sigmoid function, the logistic function $\sigma(z) = 1/(1+e^{-z})$, as '*the* sigmoid function'.

stable; in the vanishing gradient problem, conversely, the norm of the gradient decreases fast to zero, which makes learning longer dependencies harder.

2. **LSTM** LSTMs address the vanishing gradient problem commonly found in RNNs by incorporating gating functions into their state dynamics (Hochreiter and Schmidhuber, 1997). At each time step, an LSTM maintains a hidden vector h and a memory vector m responsible for controlling state updates and outputs. More concretely, the hidden vector at time step t is often computed using the formulation of Gers et al. (1999):

$$\begin{aligned} g_t^i &= \sigma(A^i h_{t-1} + B^i x_t) \\ g_t^f &= \sigma(A^f h_{t-1} + B^i x_t) \\ g_t^o &= \sigma(A^o h_{t-1} + B^i x_t) \\ \tilde{c}_t &= \tanh(A^c h_{t-1} + B^c x_t) \\ c_t &= g_t^f \odot c_{t-1} + g_t^i \odot \tilde{c}_t \\ h_t &= g_t^o \odot \tanh(c_t) \end{aligned} \tag{2.17}$$

This typical formulation employs three gates: the input gate (g_t^i), forget gate (g_t^f), and output gate (g_t^o). Variants of this formulation also exist, such as LSTMs with peephole connections (Gers and Schmidhuber, 2001; Gers et al., 2002), etc. An introduction on the use of neural networks, including LSTMs, in NLP can be found in Goldberg (2016).

Output Softmax Layer The output probability is estimated using the softmax function, i.e.

$$p(s_t|h_t) = \text{softmax}(\psi(s_t)) \tag{2.18}$$

where $\psi(.)$ is a score function. This is similar to the output layer of a multi-class softmax classifier.

Training Neural LMs are typically trained to minimise a cross-entropy objective on the training corpus:

$$\mathscr{L}_{train} = -\frac{1}{N} \sum_{s_t \in train} \ln p(s_t|s_{<t}) \tag{2.19}$$

Evaluation Language models can be evaluated as follows:

- **Perplexity** A common performance metric for LMs is per token perplexity (Eq. 2.20), evaluated on a test corpus. It can also be interpreted as the average branching factor of a LM, i.e. the size of an equally weighted distribution (e.g. number of sides on a fair die) that would give the same uncertainty as the estimated distribution. This simple interpretation of perplexity and its independence from the choice of a logarithm basis makes it a more attractive evaluation metric than cross-entropy.

$$PP_{test} = \exp(\mathscr{H}_{test}) \tag{2.20}$$

  When OOV symbols (e.g. 'UNK') are included in the corpus, perplexity is sensitive to out-of-vocabulary (OOV) rate. This complicates the direct comparison of perplexities between different datasets and between models with different vocabulary sizes. As an extreme example, in a document where all words are out of vocabulary, the best perplexity is achieved by a trivial model that predicts everything as unknown.

- **Adjusted Perplexity** Ueberla (1994) proposed Adjusted Perplexity (APP; Eq. 2.22), also known as unknown-penalised perplexity (Ahn et al., 2016), to cancel the effect of the out-of-vocabulary rate on perplexity. The APP is the perplexity of an adjusted model that uniformly redistributes the probability of each out-of-vocabulary class over all different types in that class:

$$p'(s) = \begin{cases} p(s)\frac{1}{|OOV_c|} & \text{if } s \in OOV_c \\ p(s) & \text{otherwise} \end{cases} \tag{2.21}$$

  where OOV_c is an out-of-vocabulary class (e.g. words and numerals), and $|OOV_c|$ is the cardinality of each OOV set. Equivalently, adjusted perplexity

can be calculated as:

$$\begin{aligned} APP_{test} &= \exp\left(\mathscr{H}_{test} + \sum_{c} \mathscr{H}^{c}_{\text{adjust}}\right) \\ \mathscr{H}^{c}_{\text{adjust}} &= -\sum_{t} \frac{|s_t \in OOV_c|}{N} \log \frac{1}{|OOV_c|} \end{aligned} \tag{2.22}$$

where N is the total number of tokens in the test set and $|s \in OOV_c|$ is the count of tokens from the test set belonging in each OOV set.

- **Task-specific Evaluation** Perplexity-based metrics are generic language modelling evaluation metrics that may or may not correlate with the performance in a downstream task. Therefore, a task-specific evaluation is often required.

2.1.3 Text Prediction

Text prediction is a family of tasks that predict text from a context. Two tasks from this family are word prediction and word completion.

2.1.3.1 Word Prediction

Task Definition *Word prediction* is the task of proposing a ranked list of suggestions for each next word in a text given a context. Typically, a language model is used to accumulate the history of the already typed words into the context and to make the suggestions.

Evaluation The word prediction task can be evaluated with the following metrics:

- **Precision and Recall at Rank** For ranked outputs precision and recall are calculated at a rank r by only considering the top r retrieved items in the list for each instance/query q_i:

$$Precision_i@r = \frac{\sum_{k=1}^{r} rel(a_{i,k})}{r} \tag{2.23}$$

$$Recall_i@r = \frac{\sum_{k=1}^{r} rel(a_{i,k})}{\sum_{k=1}^{|L_i|} rel(a_{i,k})} \tag{2.24}$$

where $rel(.)$ is a relevance function that assigns 1 to relevant and 0 to non-relevant items. The performance of the system is summarised by reporting the metrics averaged over all queries/instances. Precision and recall at rank ignore items ranked lower in the list than rank r.

- **Mean Reciprocal Rank** Mean reciprocal rank is a metric that emphasises the single highest-ranked relevant item in the list. It is the mean of the reciprocal rank for all queries/instances, where the reciprocal rank for a single query/instance q_i is the multiplicative inverse of the rank of the first relevant item:

$$RR_i = \begin{cases} 0 & \text{no relevant in results} \\ \frac{1}{r^*} & \text{otherwise} \end{cases} \tag{2.25}$$

where r^* is the first rank where a relevant item is found in the list.

2.1.3.2 Word Completion

Task Definition *Word completion* is the task of predicting the next word in a text, as the user is typing. If the correct word is suggested, the user can auto-complete the word by pressing a single key, e.g. the tab key, which saves a few keystrokes; otherwise, the user continues typing characters until the intended word is suggested or until the user finishes typing the word.

Evaluation Word completion has task-specific evaluation metrics that compare the aided and unaided streams of keys:

- **Keystroke Savings and Unnecessary Distractions** Keystroke Savings (KS) measures the percentage reduction in keys pressed compared to character-by-character text entry and Unnecessary Distraction (UD) measures the average number of unaccepted character suggestions that the user has to scan before

completing a word:

$$KS = \frac{keys_{unaided} - keys_{aided}}{keys_{unaided}} \tag{2.26}$$

$$UD = \frac{count(suggested, not\ accepted)}{count(accepted)} \tag{2.27}$$

- **Precision and Recall** Alternatively, one can view word completion as a classification or retrieval task for each next character and evaluate using precision and recall:

$$Precision = \frac{count(accepted)}{count(suggested)} \tag{2.28}$$

$$Recall = \frac{count(accepted)}{count(total\ characters)} \tag{2.29}$$

Both sets of evaluation metrics for word completion are equivalent, as Bickel et al. (2005) note that UD and KS correspond to these precision and recall metrics, respectively:

$$Precision = \frac{1}{1+UD}, \quad Recall = KS \tag{2.30}$$

2.1.4 Error Detection and Correction

Error detection and correction are the tasks of detecting and correcting, respectively, errors in texts. There are two main types of text errors:

1. **Grammatical Error** This is a violation of the grammatical rules of a language, e.g. 'I eats an apple'.

2. **Semantic Error** This is a violation of the rules of meaning of a language, e.g. 'An apple eats me'.

2.1.4.1 Error Detection

Task Definition *Error detection* is the binary classification task of deciding whether a text contains an error or not. For example, the sentence *'I eats an apple'* should trigger positive detection, as it contains an error (grammatical error). Error detection can also be viewed as a binary retrieval task, where we want to retrieve all texts that contain errors from a collection of texts.

Evaluation Error detection can be evaluated with the following metrics:

- **Precision and Recall** These are defined as in binary classification or, equivalently, as in retrieval:

$$Precision = \frac{|relevant \cap retrieved|}{|retrieved|} \tag{2.31}$$

$$Recall = \frac{|relevant \cap retrieved|}{|relevant|} \tag{2.32}$$

  where the sets contain the retrieved items and the relevant items.

- **F-score** This metric summarises both precision and recall. The F1 metric can be used if precision and recall are equally important. In some error detection tasks, precision is more important to minimise nuisance to the user and a different F-score is used, e.g. the F0.5.

2.1.4.2 Error Correction

Task Definition *Error correction* is the task of proposing a set of edits to correct any errors in a given text. For example, the grammatical error in the sentence *'I eats an apple'* can be corrected by applying either of the edits $\{eats \rightarrow eat\}$ or $\{I \rightarrow She\}$.

Evaluation Error correction can be evaluated as follows:

- **Precision and Recall** These are defined as in retrieval:

$$Precision = \frac{\sum_{i=1}^{N} |e_i \cap g_i|}{\sum_{i=1}^{N} |e_i|} \tag{2.33}$$

$$Recall = \frac{\sum_{i=1}^{N} |e_i \cap g_i|}{\sum_{i=1}^{N} |g_i|} \tag{2.34}$$

where e_i is the set of proposed edits and g_i is the set of ground truth edits.

- **F-score** The F1 or F0.5 scores are used to summarise precision and recall.

2.2 Numerical Variables and Related Tasks

A numerical variable is a variable whose value is a number. There are two types of numerical variables:

- **Discrete Variables** These variables take their value from a countable set (finite or countably infinite), e.g. roll of a die (set $\{1,2,3,4,5,6\}$), number of children (set of natural numbers), valences of chemical elements (set of integer numbers), etc.

- **Continuous Variables** These variables take their value from an uncountably infinite set, e.g. someone's age, height, credit card balance (all from the set of real numbers), etc.

Probability Functions Depending on the type of the numerical variable, its probability may be defined by one of the following probability functions:

- **Probability Mass Function** For a discrete random variable X, the probability that it takes on a particular value x is known as the probability mass function (PMF), $p(x) = P(X = x)$, similarly to categorical variables.

- **Probability Density Function** For a continuous random variable X, the probability that it takes on any particular value x is always exactly equal to zero, i.e. $P(X = x) = 0 \forall x$, since there infinite possible values. For this reason, we are instead usually interested in the probability that X belongs in an interval, $P(a < X \leq b)$. This is estimated with the help of a probability density function (PDF), $p(x)$, as $P(a < X \leq b) = \int_a^b p(x)dx$.

- **Cumulative Density Function** Since discrete and continuous variables call for probability mass and density functions, respectively, it is sometimes more convenient to instead work with the cumulative density function (CDF), $F(x) = P(X \leq x)$. This is defined as $F(x) = \sum_{t \leq x} p(t)$ and $F(x) = \int_{-\infty}^{x} p(t)dt$ for discrete and continuous variables, respectively, and is a non-decreasing function. Now, for both types, the interval probability is computed as $P(a < X \leq b) = F(b) - F(a)$.

Mixed-Type Variables The CDF makes it possible to consider and analyse a third type of numerical variable: *mixed random variables*, which are numerical random variables that are neither discrete nor continuous. For example, the amount of rainfall within a 24-hour period is not purely continuous, because the existence of dry days implies that $P(rainfall = x) > 0$ for $x = 0$, nor is it purely discrete, because the set of all possible values is not countable. Mixed type variables occur naturally when a continuous random variable is truncated, e.g. through rounding.

2.2.1 Regression

Task Definition *Regression* is the problem of predicting the value of a numerical attribute of interest for an object based on features of the object, e.g. predict a patient's risk of disease based on their demographic information. More formally, a regression problem is a supervised learning problem, where the target variable is a numerical variable. Regression with textual features is known as text regression, e.g. predict a movie's revenue from its reviews.

Linear Regression Given a data set $\{(xi, yi)\}$, a linear regression model assumes that the relationship between the input variable x_i and output variable y_i is linear:

$$y_i = w^T x_i + \varepsilon_i \tag{2.35}$$

where ε_i is an unobserved random variable that adds noise. Assuming that the noise variable is iid normally distributed, i.e. $\varepsilon \sim N\left(0, \sigma^2\right)$, the probability of the output

given the input becomes:

$$p(y|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}(y - w^T x)^2\right) \tag{2.36}$$

Maximising the likelihood yield the maximum likelihood estimator (MLE) for the model:

$$w_{MLE} = \underset{w}{\operatorname{argmax}} \ln(L) = -\frac{1}{2\sigma^2}(y - w^T x)^2 \tag{2.37}$$

Evaluation Let $\hat{y}_t$ be the prediction of the model, and let v_i be the ground truth value. Regression can be evaluated with the following metrics:

- **Absolute Errors** In reverse order of tolerance to extreme errors, some of the most popular evaluation metrics from the regression literature are Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Median Absolute Error (MdAE):

$$\begin{aligned} e_i &= y_i - \hat{y}_i \\ RMSE &= \sqrt{\frac{1}{N}\sum_{i=1}^{N} e_i^2} \\ MAE &= \frac{1}{N}\sum_{i=1}^{N} |e_i| \\ MdAE &= \text{median}\{|e_i|\} \end{aligned} \tag{2.38}$$

  These metrics depend on the scale and units of the data.

- **Relative Errors** If the data contains values from different scales, relative metrics are often preferred, such as the Mean Absolute Percentage Error (MAPE):

$$\begin{aligned} p_i &= \frac{y_i - \hat{y}_i}{v_i} \\ MAPE &= \frac{1}{N}\sum_{i=1}^{N} |p_i| \end{aligned} \tag{2.39}$$

2.3 Words and Numbers

This section provides some additional information on the topics of words, numbers, and numerals.

Words A word is a symbol that expresses a meaning (object/idea) in a language. e.g. 'word', and 'antidisestablishmentarianism' are words in the English language, but 'wrod' and 'mentarianism' are not.

Characters Characters are basic symbols used to construct words, e.g. the word 'word' is constructed from characters 'w', 'o', 'r', and 'd'. For example, the characters of the English language are the lowercase and uppercase forms of the 26 letters of the alphabet, numeric digits, punctuation marks, and a variety of other symbols, e.g. the ampersand ('&'), the at sign ('@'), currency symbols (£, €, $, ...), etc.

Numbers A number is an abstract, mathematical object with a specific magnitude. Examples of numbers include the natural numbers such as 1 and 2, integer numbers such as 2018 and -3, rational numbers such as $1/2$ and $-3/4$, real numbers such as $\sqrt{2}$ and π, complex numbers such as $\sqrt{-1}$ and $2+3i$, etc. Numbers are used to count, measure, and label.

Numerals A numeral is a symbol used to represent a number, e.g. the numeral '5' represents the integer number five. One could choose alternative symbolic notations to represent the same number, e.g. 五, 101, V, ε', |||||, etc.

Digits Digits are basic symbols used to construct numerals, e.g. the numeral '2018' is constructed from the digits '2', '0', '1', and '8'. Single digits typically represent small numbers.

Numeral Systems A numeral system is a set of digits together with rules for arranging digits into numerals that represent numbers. The Hindu-Arabic numeral system, the world's most common numeral system, uses 0-9 and the decimal point as its set of digits. Under the system's rules, '3.14159' is a well-formed numeral, but '3.141.59' is not (valid numerals contain at most one decimal point). The intended numeral system needs to be known to unambiguously interpret a numeral, as the same numeral can represent different numbers in different numeral systems. For example, numeral '101' represents number 101 in the decimal system and number 5 in the binary system.

Types of Numeral Systems The Hindu-Arabic system is a positional or place-value system, i.e. the value of a digit changes depending on its position in the numeral

string. For example, digit 5 can mean five, fifty, five hundred, etc., depending on its placement as the first, second, third, etc., position before the decimal point (or end of numeral), respectively. Contrast that with the Roman numeral system, a non-positional system, where V always denotes five and different digits are employed for fifty (L), five hundred (D), etc.

Bases of Numeral Systems The number of unique digits, including zero, that a positional numeral system uses is called its base or radix. The Hindu-Arabic numeral system is a decimal or base-10 system, i.e. it uses 10 digits. Non-decimal systems are occasionally preferred by different cultures or in specialised applications: unary (base-1) systems are used in counting with tally marks; binary (base-2) in digital computing; vigesimal (base-20) in some languages currently or historically, e.g. ‘quatre-vingts’, ‘laurogei’, ‘fourscore’, and ‘kan k’áal’ for 80 in French, Basque, English, and Yucatec Mayan, respectively, literally mean ”four-twenties”; sexagesimal (base-60) in ancient Summer and Babylonia and in modern circular coordinate and time measuring systems.

Chapter 3

Related Work

It's everywhere: dates, times, license plate
numbers, pages of books, even elevator floor lights.'
— Jim Carrey as 'Walter Sparrow',
in *The Number 23*

This chapter positions our work within the broader literature by discussing related work.

Structure of Chapter This chapter is structured as follows:

- In Section 3.1, we present existing work related to language modelling.
- In Section 3.2, we present existing work related to text prediction (word prediction and completion).
- In Section 3.3, we present existing work related to error detection and correction.
- In Section 3.4, we present existing work related to numbers in natural language processing.
- In Section 3.5, we present applications of natural language processing with an emphasis in the clinical domain.

3.1 Language Modelling

Language Models Language Models (LMs) are statistical models that assign a probability over sequences of words. Language models have been used to ad-

dress a wide range of tasks either as components in the noisy channel paradigm, e.g. for part of speech tagging (Church, 1989), machine translation (Brown et al., 1993), speech recognition (Jelinek, 1997), information retrieval (Berger and Lafferty, 2017), text summarisation (Knight and Marcu, 2002), and question answering (Quirk et al., 2004); or, more recently, as end-to-end conditional language models, e.g. for speech recognition (Mikolov et al., 2010; Zhang et al., 2017; Xiong et al., 2017; Prabhavalkar et al., 2017), machine translation (Luong et al., 2015; Gülçehre et al., 2017; Gao et al., 2017), text summarisation (Filippova et al., 2015; Gambhir and Gupta, 2017), and question answering (Wang et al., 2017).

Approaches Modelling approaches to language models have evolved as follows:

1. **N-grams** Early statistical language models were estimated by counting n-grams, i.e. contiguous sequences of n words in the text, from a training set. This approach suffers from sparsity, as unobserved n-grams are associated with zero probabilities, which can be alleviated through smoothing techniques (Chen and Goodman, 1999; Kneser and Ney, 1993).

2. **Feed-Forward Neural Networks** Neural approaches enforce smoothness by using projections to lower dimensional spaces and non-linear functions. Early neural LMs were feed-forward networks with a fixed-length word history window as input and the next word as output (Bengio et al., 2003).

3. **Recurrent Neural Networks** More recently, Recurrent Neural Networks (RNNs) have allowed LMs to learn longer-range dependencies by accumulating a running representation of the complete history, instead of using a fixed-length history (Mikolov et al., 2010). This approach can suffer from the vanishing gradient problem, which can be addressed by using more sophisticated recurrent units, such as the Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014) networks.

Conditional Language Models Conditional LMs compute the probability of a text given various types of contexts, e.g. document in a foreign language for machine

translation (Cho et al., 2014), longer documents for summarisation (Nallapati et al., 2016), speech for speech-to-text (Graves and Jaitly, 2014), images (Vinyals et al., 2015b; Donahue et al., 2015) or videos (Yao et al., 2015) for captioning, and others. The design of conditional LMs often makes use of the encoder-decoder framework (Cho et al., 2014), where a dedicated component, the encoder, builds the representation of the context which is then used to influence the hidden state of another neural network, the decoder. Sequence-to-sequence learning (Sutskever et al., 2014) is a special case, where the input of the encoder and output of the decoder are sequences.

Grounded Language Models The symbol grounding problem (Harnad, 1990) refers to how a symbol, e.g. a word, derives its semantic meaning. Similarly to how the distributional hypothesis (Harris, 1954; Firth, 1957) exploits the co-occurrence between words to build semantic representations, grounded semantic representations can be built by exploiting the co-occurrence of words and non-linguistic contexts, such as images (Bruni et al., 2014; Silberer and Lapata, 2014), audio (Kiela and Clark, 2015), video (Fleischman and Roy, 2008), colours (McMahan and Stone, 2015), and smells (Kiela et al., 2015).

Gradable Words Gradable words (Sapir, 1944) denote different degrees of quality, e.g. 'tiny', 'short', 'tall', and 'gigantic' denote different degrees of height. Gradable adjectives can be interpreted as measure functions (Bartsch, 1975; Kennedy, 1999) associated with features that are easily observable (e.g. 'tall', 'old', 'hot') or more abstract (e.g. 'beautiful', 'intelligent', 'energetic'). Other parts of speech can also be gradable, such as verbs (e.g. 'to cool', 'to freeze'), nouns ('idiot', 'genius'), and adverbs (e.g., 'fairly', 'utterly') (Shivade et al., 2015; Ruppenhofer et al., 2015). Gradability expresses vagueness and uncertainty, and the semantics of gradable words are context sensitive (Kennedy, 2007; Frazier et al., 2008), e.g. in 'John is a tall boy' and 'John is a tall basketball player' the word 'tall' alludes to different measurements of height. Gradable words can be considered grounded in their associated quality, e.g. Shivade et al. (2016) use a simple probabilistic model to ground gradable adjectives from patient health records in clinical numerical measurements.

Structured Data to Text A standard problem in natural language generation (NLG) (Kukich, 1983; Holmes-Higgin, 1994; Reiter and Dale, 1997) involves reading structured data, such as tables, knowledge bases, or ontologies, and producing text that describes the data. Benchmark datasets have been created for several domains, such as generating weather forecast reports from meteorological data, e.g. WeatherGov (Liang et al., 2009) and Sumtime (Sripada et al., 2003a), generating sportcasting descriptions from sport data, e.g. Robocup (Chen and Mooney, 2008), RotoWire and SBNation (Wiseman et al., 2017), generating biographical descriptions from infoboxes e.g. WikiBio (Lebret et al., 2016), generating restaurant descriptions from metadata (Novikova et al., 2017), etc. The data that the texts are based on often contain numbers, e.g. temperatures and wind speeds, points scored, dates, ratings, etc.

Out-of-Vocabulary Words and The Unknown Word Problem In language modelling, generating rare or unknown words has been a challenge, similar to our unknown numeral problem. Gulcehre et al. (2016) and Gu et al. (2016) adopted pointer networks (Vinyals et al., 2015a) to copy unknown words from the source in translation and summarisation tasks. Merity et al. (2016) and Lebret et al. (2016) have models that copy from context sentences and from Wikipedia's info boxes, respectively. Ahn et al. (2016) proposed a LM that retrieves unknown words from facts in a knowledge graph. They draw attention to the inappropriateness of perplexity when OOV-rates are high and instead propose an adjusted perplexity metric that is equivalent to APP. Other methods that aim at speeding up LMs, such as hierarchical softmax (Morin and Bengio, 2005b) and target sampling (Jean et al., 2014), have been used to allow for increasingly larger vocabularies (Chen et al., 2015), but still suffer from the unknown word problem, as they cannot accommodate an open vocabulary. Finally, the unknown word problem does not occur when modelling with subword units, e.g. individual characters (Graves, 2013; Sutskever et al., 2011) or byte-pair encoding (BPE) (Sennrich et al., 2016), which allows for handling an open vocabulary.

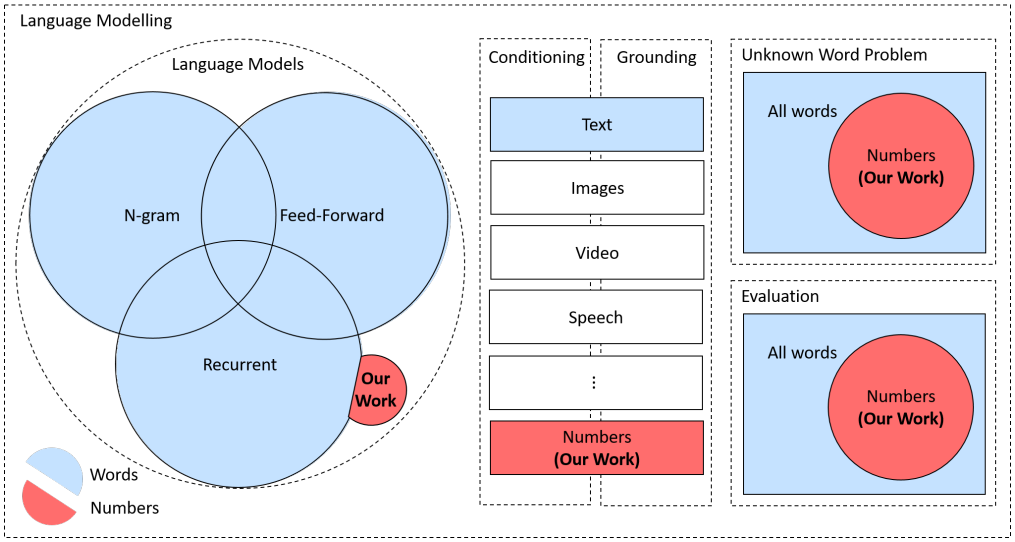

Figure 3.1: Our position among approaches to Language Modelling.

Our Work Figure 3.1 shows the position of our work among approaches to language modelling. We are extending neural language models by conditioning and grounding on a numerical modality. We address the unknown word problem for numerals, for which we provide explicit evaluations. This work is related to the structured data to natural language text generation, where the structured data comes from a entries in a knowledge base.

3.2 Text Prediction

Text Prediction Word prediction and word completion are aspects of text prediction, a collection of functionalities for assisting users at text entry tasks by anticipating the next words, phrases, or sentences that the user will produce. Text prediction originated in Augmentative and Alternative Communication (AAC) for people with motor or speech impairments (Beukelman and Mirenda, 2005; Brown, 1992). The scope of text prediction now extends to a gamut of applications, such as data entry in mobile devices (Dunlop and Crossan, 2000), interactive machine translation (Foster et al., 2002), search term auto-completion (Bast and Weber, 2006), and assisted clinical report compilation (Eng and Eisner, 2004; Cannataro et al., 2012). Moreover, word prediction has been used as a task to evaluate the natural language understanding capabilities of neural networks (Paperno et al., 2016; Ghosh et al.,

2016).

Aims The main aim of text prediction is to reduce the effort and time for text entry. Assuming text entry on a keyboard, this aim is usually achieved by reducing the number of keystrokes needed, and the performance is evaluated by as the keystroke savings: the percentage of keystrokes saved (Li and Hirst, 2005; Trnka and McCoy, 2007; Carlberger et al., 1997; Garay-Vitoria and Abascal, 2006; Newell et al., 1998). Surveys report keystroke savings between 29% and 56% (Garay-Vitoria and Abascal, 2006). Some applications of text prediction have alternative aims: improving the quality of text by preventing misspellings, promoting the adoption of standard terminologies, and allowing for exploration of new vocabulary (Sevenster and Aleksovski, 2010; Sevenster et al., 2012).

Approaches The context up to the current word is very important for correctly predicting or completing the next intended word. For example, in addition to the prefix of the current word, knowing the previous words and characters can significantly improve word completion (Fazly and Hirst, 2003; Van Den Bosch and Bogers, 2008). Most approaches to word prediction and completion use a LM to accumulate the history of already typed work into the context (Bickel et al., 2005; Wandmacher and Antoine, 2008; Trnka, 2008; Ghosh et al., 2016). Earlier work used n-gram LMs (Fazly and Hirst, 2003; Foster et al., 2002; Matiasek et al., 2002), while more recent work uses neural LMs (Ghosh et al., 2016). This context is often enhanced with additional syntactic or semantic information to improve the predictive system's performance:

- **Syntactic Context** Syntactic input features, e.g. part-of-speech tags (Fazly and Hirst, 2003), can help produce grammatically correct suggestions. Alternatively, syntactic information can help limit the search space only to grammatically appropriate words, e.g. consider only words for which the whole final sentences have valid syntactic trees (Wood and Lewis, 1996), or use syntactic parsing to automatically inflect content words allowing the user to type the shorter noninflected forms (McCoy et al., 1998). Finally, syntactic language models are language models that use syntactic structure to model

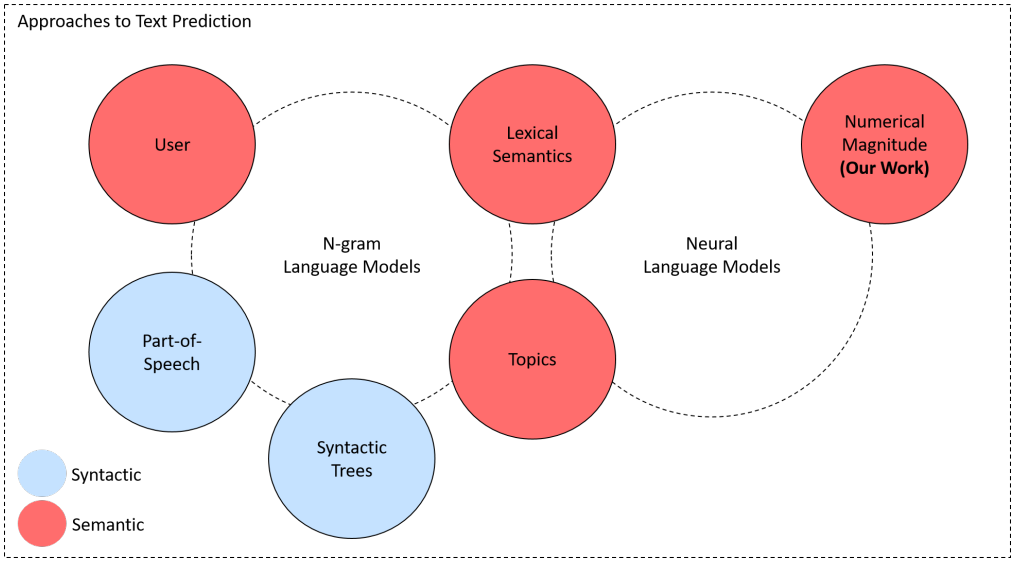

Figure 3.2: Our position among approaches to Text Prediction.

long-distance dependencies in an incremental, left-to-right manner, and can be used to predict the next word (Chelba and Jelinek, 1998, 2000; Roark, 2001; Charniak, 2001; Emami and Jelinek, 2005).

- **Semantic Context** Semantic input features, e.g. the topic of previous sentences (Ghosh et al., 2016) or word vectors that capture lexical semantics (Wandmacher and Antoine, 2008), can help produce semantically correct suggestions. Alternatively, semantic information can be incorporated through domain adaptation, e.g. through adapting the LM to the user (Wandmacher et al., 2008) or topic (Trnka, 2008).

Our Work Figure 3.3 shows the position of our work among approaches to text prediction. We are incorporating numerical magnitude into the semantic context of neural language models for text prediction.

3.3 Error Detection and Correction

Approaches to Error Detection and Correction Leacock et al. (2014) and Tetreault et al. (2014) identify the following approaches to Error Detection and Correction (EDC):

1. **Rule-Based** This approach uses hand-crafted rules to tackle specific error

types. While rule-based systems do not require training corpora, they do not generalise to errors not explicitly covered by dedicated rules.

2. **Classification** This approach uses binary classifiers to detect errors or multi-class classifiers to correct errors by recommending edits from a finite set of alternatives, the *confusion set*. Typically, a classifier is trained for each targeted error type on a training corpus with ground-truth annotations for errors and required correction edits.

3. **Language Modelling** This approach uses a language model to score the original and alternative texts, where words have been substituted with alternatives (e.g. from confusion sets). The text with the highest probability under the language model is chosen as the correct text in the given context. Language models are not restricted to correcting pre-specified error types, but they require large training corpora of correct language usage.

4. **Statistical Machine Translation** This approach views error correction as a translation from an erroneous, "noisy" language into an error-free, corrected language. The translation system is trained on a training corpus with pairs of erroneous and corrected texts. This approach is related to the language modelling approach, and many machine translation systems rely on a language model as a core component.

5. **Web-based** This approach mimics the strategy of humans who search for alternative constructions on the web and compare usage counts as a proxy of correctness. While large scale Web data are readily available, this approach can suffer from sparsity issues with certain errors, the strategy is not context dependent, and the counts can unreliably change over time.

Research in Grammatical EDC Most early work in grammatical EDC used rule-based approaches, and hand-crafted rules are still commonly used in hybrid systems for certain well understood errors, such as subject-verb agreement. Classification approaches use various features, including syntactic, lexical, and semantic

features (Gamon et al., 2008; De Felice and Pulman, 2008; Tetreault and Chodorow, 2008), and are trained on real or artificially generated errors (Felice and Yuan, 2014; Felice et al., 2014; Rozovskaya and Roth, 2010a,b). Language models have been used to detect errors that are later corrected via other approaches (Hdez and Calvo, 2014), to correct errors via scoring suggested corrections from other approaches (Boroş et al., 2014; Gamon et al., 2008; Chodorow et al., 2010; Felice et al., 2014), and in combination with classification (Gamon, 2010). Web-based approaches have been found to perform promisingly for determiner errors in English, but less so for collocation errors (Yi et al., 2008). There have been several shared tasks on grammatical EDC (Dale and Kilgarriff, 2011; Dale et al., 2012; Ng et al., 2013, 2014), and many of the participating teams used statistical machine translation approaches.

Research in Semantic EDC Many semantic errors manifest themselves as the wrong choice of content words, which is the third most frequent error type (Leacock et al., 2010). There has been substantially less work in semantic EDC than grammatical EDC, and tackling of the first is often subsumed into the latter. Rule-based approaches to semantic EDC compare usage against a database of known errors (Shei and Pain, 2000; Wible et al., 2003; Chang et al., 2008; Park et al., 2008). Such approaches do no generalise to unseen errors and are not context dependent. Wu et al. (2010) used a classifier to correct verb usage in the abstracts of academic articles. Other work detects semantically anomalous adjective-noun combinations in isolation (Vecchi et al., 2011) and in texts (Kochmar and Briscoe, 2013). Machine translation approaches have been used to correct the word choices of language learners (Dahlmeier and Ng, 2011) and to improve the output of speech recognition systems by applying semantic EDC in post-processing (Jeong et al., 2004, 2003). Semantic EDC is related to the task of writing improvement (Chang et al., 2008; Futagi et al., 2008), which involves editing a text to improve its relative quality without making claims about its absolute correctness.

Our Work Figure 3.3 shows the position of our work among approaches to error detection and correction. We are following a machine learning, language modelling

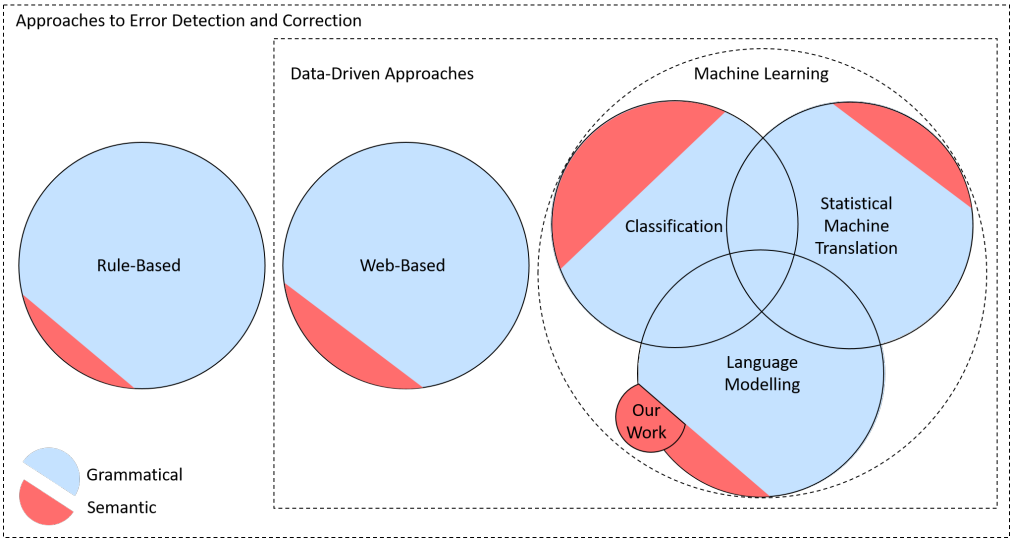

Figure 3.3: Our position among approaches to Error Detection and Correction.

approach for semantic error corrections. We extend existing approaches by using language models that are sensitive to numerical information.

3.4 Numbers in Natural Language Processing

Text Regression Text regression is the NLP task of predicting a real-valued quantity from an associated piece of text (Kogan et al., 2009). Applications of the task use text to predict financial risk (Kogan et al., 2009), movie reviews and revenues (Joshi et al., 2010), influenza rates (Lampos and Cristianini, 2010), author age (Nguyen et al., 2011), scientific article downloads and citations (Yogatama et al., 2011), restaurant reviews and menu prices (Chahuneau et al., 2012), and election results (Lampos et al., 2013). Early work uses linear regression with simple word occurrence features and metadata, while more recently deep neural networks have been employed (Bitvai and Cohn, 2015). Interpretation of the parameters of the regression models has revealed strong connections between real-world quantities and certain words or phrases. Occasionally, predicting a continuous variable has been addressed as a classification problem by discretising the continuous variable into binned categories, e.g. continuous author age binned into age ranges (Schler et al., 2006; Garera and Yarowsky, 2009). A variety of techniques have been employed to predict the geolocation coordinates of tweets on Twitter (Rahimi et al.,

2015b,a, 2017b,a; Lau et al., 2017).

Numbers in Commonsense Reasoning Numbers can be uses for commonsense reasoning, e.g. knowing the sizes of books and libraries, we can infer a content-container relation between them (Aramaki et al., 2007). A common framework for acquiring commonsense knowledge about numerical attributes is based on using templates to collect a a corpus of numerical attribute:value pairs and then modelling those attributes with a normal distribution (Aramaki et al., 2007; Davidov and Rappoport, 2010; Iftene and Moruz, 2010; Narisawa et al., 2013; de Marneffe et al., 2010).

Numbers in Question Answering Question answering systems have checked whether numerical values in their answers are within common-sense ranges, using external knowledge resources, such as a manually curated commonsense knowledge based (Chu-Carroll et al., 2003; Prager et al., 2003) or semantic type and typical numerical range of quantities (Hovy et al., 2002)

Numbers in Recognising Textual Entailment Numerical quantities have been recognised as important for recognising textual entailment (RTE) (Lev et al., 2004; Dagan et al., 2013). Roy et al. (2015) proposed a quantity entailment task that focused on whether a quantity can be inferred from a text and, if so, what its value should be. Different categories of common-sense knowledge are required for RTE, such as arithmetic knowledge (LoBue and Yates, 2011), and many system failures occur then numeric reasoning is necessary (Sammons et al., 2010).

Numbers in Information Extraction Another task that deals with numerals is numerical information extraction. The aim of the task is to extract all numerical relations, i.e. relations with numbers as arguments, from a corpus. This can range from identifying a few numerical attributes (Nguyen and Moschitti, 2011; Intxaurrondo et al., 2015) to generic numerical relation extraction (Hoffmann et al., 2010; Madaan et al., 2016). Other work focuses on algorithms and data structure for searching documents with number-range queries (Fontoura et al., 2006; Yoshida et al., 2010).

Numbers in Automated Problem Solving Much work has been done in solving arithmetic (Mitra and Baral, 2016; Hosseini et al., 2014; Roy and Roth, 2016), geometric (Seo et al., 2015), and algebraic problems (Zhou et al., 2015; Koncel-Kedziorski et al., 2015; Upadhyay et al., 2016; Upadhyay and Chang, 2016; Shi et al., 2015; Kushman et al., 2014) expressed in natural language. Such models often use mathematical background knowledge, such as linear system solvers. The output of our model is not based on such algorithmic operations, but could be extended to do so in future work.

Numbers in Natural Language Generation A standard problem in natural language generation (NLG) (Kukich, 1983; Holmes-Higgin, 1994; Reiter and Dale, 1997) involves reading structured data and producing text that describes this data. Many of the specialised domains for structured data-to-text have data sources that are number-intensive, e.g. sensor data from a gas turbine (Yu et al., 2007), sensor data from a neonatal intensive care unit (Sripada et al., 2003b), and weather forecast data (Sripada et al., 2003a) (for more examples see discussion on Structured Data to Text in Section 3.1). Chaganty and Liang (2016) automatically generates short descriptions to put numbers into perspective, e.g. '$131 million is about the cost to employ everyone in Texas over a lunch period'. Other work investigates differences in word usage between different authors, such as weather forecasts (Reiter et al., 2005).

Our Work In this thesis, we use principles found in common approaches for several tasks that use numbers: we consider models with text as input and numbers as output (text regression); we use attribute:value pairs from a knowledge base to make inferences about text (commonsense reasoning and RTE); and we predict words in a text (natural language generation). Unlike the existing work, we embed these principles into a language model. Also, we do not use mathematical background knowledge (e.g. as often in automated problem solving), but all numerical knowledge is learnt from data.

3.5 Applications of Natural Language Processing

Clinical Domain Natural language processing techniques can be applied to any text (or collection of texts) that arises from or is related to clinical activities, e.g. clinical notes and narratives, discharge summaries, clinical research papers, etc. These techniques can facilitate clinical decision support (Demner-Fushman et al., 2009), and several NLP systems and toolkits have been developed for use in the clinical domain (Savova et al., 2010; Soysal et al., 2017; Aronson, 2001).

Information Extraction Much of the NLP work in the clinical domain focuses on information extraction from the text in electronic health records, such as the extraction of categorical attributes, e.g. patient smoking status (Uzuner et al., 2008), disease status (Friedman et al., 1994), clinical codes (Stanfill et al., 2010), medication information (Uzuner et al., 2010), obesity (Uzuner, 2009), negation of attributes (Huang and Lowe, 2007), etc, or numerical attributes, e.g. temporal information (Tang et al., 2013), medication dosages (Evans et al., 1996), etc. A review of information extraction from clinical texts can be found in Meystre et al. (2008) and Wang et al. (2018).

Text Entry and Processing Various techniques and tools have been employed for assisting with the entry in electronic health records, such as assisted text entry (Cannataro et al., 2012), text standardisation through templating (Johnson et al., 2008), text completion (Eng and Eisner, 2004), etc. Other work is concerned with identifying and correcting errors in the texts of electronic health records, e.g. inconsistencies between text and structured data (Bowman, 2013), prescription errors (Singh et al., 2009), number-based errors (Arts et al., 2002), spelling mistakes (Lai et al., 2015), etc. Overall, error rates in clinical data range from 2.3% to 26.9% (Goldberg et al., 2008). Finally, other streams of work process clinical texts to generate related texts for particular applications, e.g. translations into foreign languages (Randhawa et al., 2013), customised educational messages for trainee clinicians (Denny et al., 2015), simplified clinical texts that are more understandable to patients (Green, 2006), text summaries (Afantenos et al., 2005), answers to questions (Jacquemart and Zweigenbaum, 2003), etc.

Our Work In this thesis, we apply and extend standard techniques from NLP to text data from the clinical domain. Our experiments with classification and regression share the goals of information extraction, but we approach the problem in a predictive, rather than in an extractive way. Our experiments with language modelling show that our techniques can be used to assist the entry (through text prediction) and editing (through error detection and correction) of clinical text in a simulation, rather than in an applied clinical setting.

Chapter 4

Establishing the Connection between Words and Numbers

–How long will the morphine be effective?

–No telling with the size of her body.

— Otto Waldis as 'Dr. Heinrich Von Loeb'
to Roy Gordon as 'Dr. Isaac Cushing',
in *Attack of the 50 Ft. Woman*

There is often an intuitive connection between certain numerical attributes and textual descriptions. For example, if we know that someone has a height that is greater than the average human height, we might describe that person as 'tall'. Likewise, if we know that someone has been described as 'tall', we can assume with some confidence that their height is greater than average. Similar connections exist between other sets of gradable textual terms and measurable attributes, e.g. 'light' or 'heavy' for weight, 'slow' or 'fast' for speed, 'light' or 'dark' for luminosity, etc. Learning such connections between numerical attributes and text can allow us to ground the semantics of words in measurements and perform reasoning with numerical and textual information sources in commonsense and expert domains.

Research Questions In this chapter, we will investigate the connection between words and numbers by measuring how helpful numerical attributes (e.g. measurements expressed as numbers) at predicting words from a text, and vice versa. We

frame this investigation as two research questions:

Research Question Q.1
Can inputting numbers into a model that predicts words (classifier for the next word; single-step left-to-right language modelling) improve its performance?

Research Question Q.2
Can inputting words into a model that predicts numbers (regressor) improve its performance?

Scope We will address the two research questions within the expert domain of clinical records. A clinical dataset will be used throughout this thesis, and will is described in the upcoming sections and in the Appendix. A clinical record describes information about an individual patient in one of two ways:

- **Knowledge Base (KB)** The structured part of the clinical record is a collection of attribute-value pairs that describe the patient's demographic information (e.g. age, sex, etc.) and results of medical tests (e.g. cardiac function);
- **Text** The unstructured part of the clinical record is a textual report written by a health-care professional to document further aspects of the patient's history, examination, diagnosis, and conclusion, as this information becomes available.

Numbers are represented either by the values of numerical attributes in the KB or by numerals in the text. Both the KB and text of the record are essential for the effective communication of clinical information (Lovis et al., 2000).

Connecting Words and Numbers: An Example Table 4.1 shows examples of clinical records that highlight evidence towards inferring that the patient suffers from dilated cardiomyopathy (DCM), a condition that accounts for about one in three cases of congestive heart failure (Kasper et al., 2005). In healthy individuals, the

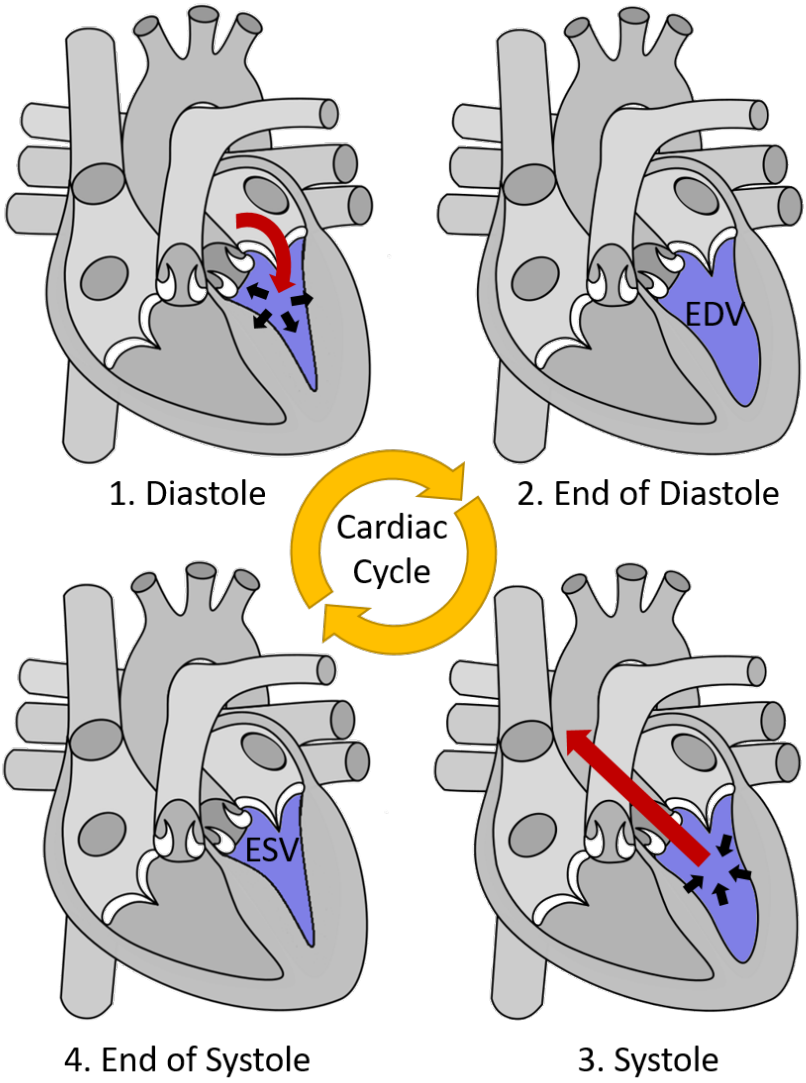

Figure 4.1: Cardiac cycle. The cycle consists of the following phases (simplified to only show changes in the volume of the left ventricle): 1. During diastole, blood enters the left ventricle (in blue) and its volume increases; 2. At the end of diastole, the ventricle has reached its End-Diastolic Volume (EDV); 3. During systole, blood exits the ventricle and its volume decreases; 4. At the end of systole, the ventricle has reached its End-Systolic Volume (ESV). The ejection fraction (EF) is calculated as $\mathrm{EF} = (\mathrm{EDV} - \mathrm{ESV})/\mathrm{EDV} \times 100\%$.

Table 4.1: Example of data from the clinical domain. Clinical records store information about patients in structured (knowledge base; KB) and unstructured (text) ways. Evidence for left ventricular dilation is **in bold** (LV EDV is left-ventricular end-diastolic volume); evidence for systolic function impairment is underlined (LV EF is left-ventricular end-systolic volume); the remaining attributes (e.g. LV ESV) are not required for understanding the example (for a description of all attributes in the KB, see Appendix).

KB	TEXT
age: 76 **sex: male** **LV EDV: 378.85** LV ESV: 279.83 LV SV: 99 LV EF: 26.1	Left Ventricular Functional Analysis Results **End diastolic volume 378.85 ml** End systolic volume 279.83 ml Stroke volume 99.0 ml Ejection fraction 26.1 % History: 76 year old man. **Dilated LV** of unknown aetiology. Unobstructed coronary arteries. Thoracic anatomy: Normal arrangement of cardiac chambers and great vessels. [...] **The left ventricle is severely dilated;** systolic function is severely impaired. [...] Conclusion: **Severe LV dilation** and systolic impairment. Likely small lateral MI. Mild eccentric AR. No perfusion defects. The appearances are **consistent with dilated cardiomyopathy** and a small lateral MI.
age: 26 **sex: male** **LV EDV: 170.16** LV ESV: 88.69 LV SV: 81.5 LV EF: 47.9 LV2 EDV: 113.4	Normal arrangement of the great vessels and cardiac chambers. Resting Cardiac Function: **LV is non-dilated in diastole and mildly dilated in systole** with mild LV systolic impairment (see above). [...] There is mild biatrial enlargement (LA dimension is 42mm). The RV size and function are within normal limits. [...] Conclusion: Mild atrial dilatation with preserved RV function. **There is mild LV dilatation** with mild LV impairment. There is no evidence of infarction or fibrosis. these findings are most **consistent with previous myocarditis or DCM** depending on the clinical context.

main pumping chamber of the heart, the left ventricle, expands and contracts to circulate blood during the cardiac cycle, as shown in Figure 4.1. In patients with DCM, i) the left ventricle has become enlarged and ii) the heart cannot pump blood efficiently (Maron et al., 2006; Brandenburg et al., 1981).[1] These two criteria can be respectively linked to the following numerical attributes:

i) the End-Diastolic Volume (EDV) measures the volume of the ventricle before

[1] More technically, i) the ventricle is dilated and ii) the systolic function is impaired.

it contracts, which characterises the size of the ventricle;

ii) the Ejection Fraction (EF) measures the percentage of blood leaving the ventricle each time it contracts, which characterises the pumping efficiency of the heart. The systolic function is said to be impaired for lower values of the EF, e.g. EF$<35\%$.

Different pathways can be used to make inferences about the patient's condition. For example, the ventricle can be confirmed to be dilated if:

- the KB contains an attribute for the EDV with a numerical value that is higher than expected for the given demographic (age, sex, and body surface area);
- the text contains certain phrases, e.g. 'the left ventricle is severely dilated', or numerals that indicate increased ventricular dimensions.

Similarly, the impairment of the systolic function can be inferred from phrases such as 'impaired systolic function' in the text or lower values of the EF in the KB or in the text. The complete list of attributes from the clinical dataset can be found in the Appendix.

Structure of Chapter This chapter is structured as follows:

- Section 4.1 investigates research question Q.1 by comparing the performances of models with textual and non-textual features in a word prediction classification task.
- Section 4.2 investigates research question Q.2 by comparing the performances of models with textual and non-textual features in a number prediction regression task.

This chapter is partly based on work previously published by the author (Spithourakis et al., 2014).

4.1 Classification: From Numbers to Words

When someone composes a text, their choice of words can be affected by textual and numerical information. For example, when a clinician composes a report about

a patient, each word can depend on other words (what has been written so far) and on the numerical attributes in the patient's KB, respectively (for a more detailed description of the clinical workflow, see the Appendix). In this section, we investigate the importance of numerical information for predicting words from clinical texts. In particular, we are interested in models that predict the next word chosen by the clinician to characterise the degree of ventricular dilation of a patient, which is typically a gradable adjective or adverb e.g. '*no* dilation', '*mild* dilation', or '*severely* dilated'. We approach this task as a single step of left-to-right language modelling, and we do not use any information on the right of the target token (i.e. we ignore matched patterns used as annotations). We explore other word prediction tasks and full-scale language modelling in Sections 5 and 6.

4.1.1 Approach

We address the problem as a multi-class classification task for word prediction with the following specifications:

- **Input** The input features can be either textual features from the text or (mostly) numerical features from the KB;
- **Output** The output class represents the word used by the clinicians to characterise the degree of ventricular dilation, e.g. 'no', 'mild', or 'severe'.

We will compare the performance of classifiers that use various combinations of textual and numerical input features.

4.1.2 Data

Provenance and Description Our dataset comprises 16,015 anonymised clinical records from the London Chest Hospital. The texts have an average length of 207 words, and the KB has the following attributes for each patient:

- age, which is a numerical attribute;
- sex, which is a categorical attribute that can take the value 'male' or 'female';
- end diastolic volume (EDV) for the left ventricle, which is a numerical attribute;

Table 4.2: Common text patterns for LV dilation. [is] = [appears] = [seems] = [appears to be] = [seems to be] = [remains], etc.; [left ventricle] = [LV] = [LV cavity] = [left ventricular], etc.

	Patterns
1.	[left ventricle] [is] [*modifier*] [dilated]
2.	[*modifier*] [dilated] [left ventricle]
3.	[*modifier*] [dilation] [of the] [left ventricle]
4.	[*modifier*] [left ventricle] [dilation]

Table 4.3: Statistics of document annotation.

class label	**#docs**	**%docs**
unmatched	4721	29.48
no	9116	56.92
borderline	16	0.10
mild	361	2.25
moderate	25	0.16
significant	25	0.16
marked	14	0.09
gross	80	0.50
severe	197	1.23
other	1460	9.12
total	16015	100.00

- ejection fraction (EF) for the left ventricle, which is a numerical attribute.

More information about the clinical dataset, e.g. descriptive statistics and the clinical workflow through which the data were acquired, can be found in the Appendix.

Pattern-Based Annotation We use a pattern matching approach to extract the degree of dilation reported by the clinician in the text. First, we examine the data to manually compile a list of patterns found in the texts to describe degrees of dilation. Table 4.2 shows these patterns. Then, we match all reports against these patterns to identify all possible modifiers of the dilation status. From these modifiers, we create the following normalised, mutually exclusive class labels:

- **unmatched**: a class for no pattern match;
- **no**: a class that groups all negative terms, i.e. ‘un’, ‘non’, ‘no’, and ‘not’;

Table 4.4: Statistics of attributes missing from the KB.

class label	EF %missing	EDV %missing
unmatched	47.81	47.77
no	36.89	36.87
borderline	12.50	12.50
mild	13.57	13.85
moderate	4.00	4.00
significant	40.00	40.00
marked	21.43	21.43
gross	33.75	33.75
severe	3.55	3.55
other	17.88	17.88
overall	37.34	37.32

- **borderline, mild, moderate, significant, marked, gross, and severe**: classes that group the most frequent terms with the same stems, e.g. 'mild' and 'mildly';

- **other**: a class for infrequent or unspecified terms, e.g. 'slight', 'extreme', 'very', no term, etc.

Table 4.3 shows statistics for our automated annotation. The dataset is imbalanced, with the majority of instances (56.92%) being assigned the 'no' label. A 29.48% of the text did not match any of the patterns and where assigned to the 'unmatched' class, and a 9.12% of them were labelled as 'other'. The remaining instances were variously distributed to the rest of the classes.

Preprocessing We preprocess the dataset following the steps below:

1. **Defining data folds** We randomly split the data into training, development and test sets using a 70/20/10 ratio, respectively.

2. **Truncating texts** We only keep the text on the left of the matched pattern. We remove the matched pattern and all text following it, so that the models do not have access to the ground truth information for the target classes, which can be trivially inferred from the matched pattern. This truncation will allow us

to assess how well the models can compose text in a left-to-right manner, i.e. similar to a single step of a language model.

3. **Tokenisation** We tokenise at the whitespaces and all punctuation (with the exception of the decimal point separator, to avoid splitting numerals). Since our task can be seen as single-step language modelling, we do not remove the stopwords.

4. **Lowercasing** We convert all characters to lowercase.

5. **Filtering out missing data** We observe that values from the KB are frequently missing. Table 4.4 shows that the EF and EDV attributes are missing in about 37% of the overall data. The percentage of missing values varies among classes, from 3.55% for the 'severe' class to 40.00% for the 'significant' class and about 47.8% for the 'unmatched' class. We filter out all instances with any missing value in the KB, which leaves us with $10,024$ instances in total. Even though the removal of instances with missing data might skew the data from their natural distribution (due to non-random missingness), the results will still be valid for that subset of the data. We did not attempt to remove outliers, so as not to further disturb the data distribution.

6. **Filtering out uninformative labels** We filter out instances belonging in the unmatched and other class classes, which leaves us with $6,364$ instances. We keep all remaining classes from Table 4.3, and we do not further merge them (other than the initial merger between adjective and adverb forms, e.g. 'mild' and 'mildly'). The classes themselves are mutually exclusive, as the clinician can only choose to write one of those descriptors in their report; however, the corresponding numerical ranges of the EF and EDV attributes overlap (see next paragraph), which makes the task non-trivial.

Statistical Description Table 4.5 shows descriptive statistics for the Ejection Fraction (EF) and End-Diastolic Volume (EDV) attributes from the Knowledge Base (KB). We report the means and standard deviations for each class and overall. The

Table 4.5: Statistics of EDV and EF attributes from the training set.

	EF		EDV	
class label	**mean**	**stdev**	**mean**	**stdev**
unmatched	61.54	11.87	144.91	47.44
no	59.59	11.54	144.49	34.79
borderline	54.60	8.14	202.40	23.64
mild	48.70	10.19	214.04	44.10
moderate	38.97	11.02	226.23	24.81
significant	39.55	11.96	287.06	34.14
marked	28.79	13.76	336.56	33.96
gross	22.88	8.09	338.22	91.62
severe	24.93	10.22	337.50	78.40
other	34.62	12.08	249.74	53.53
overall	55.79	15.23	164.67	62.27

numerical ranges of each class overlap for each of the EF and EDV attributes. The high proportion of the 'no' class in the dataset lowers the overall mean and, possibly, the mean for the unmatched texts. The standard deviation within most classes is lower than in the overall dataset, with the exception of the 'gross' and 'severe' classes for the EF attribute.

4.1.3 Experimental Setup

Models We define classifiers to predict the label class from different sets of input features:

1. **kb** This is a log-linear model that uses as input features the values of attributes from the KB (age, sex, EDV, and EF);

2. **text** This is a log-linear model that uses n-gram features (all 1-grams and 2-grams that appear at least 3 times in the texts of the training data) extracted from the texts up to the matched pattern (not inclusive);

3. **text**$\cup$**kb** This is a log-linear model that uses the combined input features from both previous models.

Training Details All models are trained with the LBFGS optimiser to minimise a cross entropy objective. To account for the class imbalance in the dataset, the

Table 4.6: Results for word prediction. Accuracy is equivalent to micro-averaged Precision (R), Recall (R), and F1. Best results in **bold**.

	micro P/R/F1	macro			weighted		
model	**Accuracy**	**P**	**R**	**F1**	**P**	**R**	**F1**
kb	69.50	28.42	25.54	22.24	**93.33**	69.50	79.13
text	87.94	33.12	26.06	28.51	86.96	87.94	87.42
text∪kb	**88.10**	**41.24**	**49.54**	**43.02**	93.28	**88.10**	**90.08**

contribution of each instance to the objective is weighed proportionally to its class frequency. All models are regularised with L2 regularisation, where the hyper-parameter is selected to optimise the training objective on the development set.

4.1.4 Results

Classification Results Table 4.6 shows classification evaluation metrics (accuracy, precision, recall, and F1) with different ways to average across classes (micro, macro or weighted) for the various models. All results are evaluated on the test set. For most metrics and manners of aggregation, the *text∪kb* model that combines textual and non-textual features achieves the best performance. The only exception is weighted precision, for which the *kb* model has the best performance, closely followed by the *text∪kb* model.

Breakdown of Classification Results Table 4.7 shows a breakdown of the classification evaluation metrics for each individual class. For the classes with few instances ('borderline', 'significant', 'moderate' and 'marked', all with less than 25 instances; Table 4.3), the performance metrics are zero for various models. Particularly for the 'borderline' and 'significant' classes, all models fail to correctly classify any instance. This partially explains why macro-averaged metrics in Table 4.6 were lower than the respective micro-averaged and weighted-averaged metrics. For the 'mild', 'gross', and 'severe' classes, the *text∪kb* has the best performance. For the 'no' class (majority class; Table 4.3), the *kb* model achieves the best precision and the *text* model achieves the best recall. Overall, the *text∪kb* model has the best F1 score for most classes.

Table 4.7: Breakdown of results for word prediction. Best performance in **bold**.

model	no	borderline	mild	moderate	significant	marked	gross	severe
				Precision				
kb	**99.18**	0.00	28.75	5.66	0.00	0.00	18.75	75.00
text	93.56	0.00	24.56	0.00	0.00	**100.00**	12.50	34.38
text∪kb	98.57	0.00	**29.56**	**14.29**	0.00	50.00	**60.00**	**77.50**
				Recall				
kb	73.97	0.00	33.33	**42.86**	0.00	0.00	**37.50**	16.67
text	**95.11**	0.00	20.29	0.00	0.00	50.00	12.50	30.56
text∪kb	90.31	0.00	**68.12**	14.29	0.00	**100.00**	**37.50**	**86.11**
				F1				
kb	84.74	0.00	30.87	10.00	0.00	0.00	25.00	27.27
text	**94.33**	0.00	22.22	0.00	0.00	**66.67**	12.50	32.35
text∪kb	94.26	0.00	**41.23**	**14.29**	0.00	**66.67**	**46.15**	**81.58**

Confusion Matrix Figure 4.2 shows the confusion matrix for the *text∪kb* model, evaluated on the test set. The counts of the matrix have been normalised for each true class. The elements along the diagonal correspond to correct predictions, while the rest of the matrix to misclassifications. The most common errors made by the model were misclassifying 'borderline' as 'no', 'moderate' as 'mild', and 'significant' as 'severe'.

Important Features Tables 4.8 and 4.9 show the 10 most important features for the *text* and *text∪kb* models, respectively, that is, the features with the most positive and negative coefficients. It should be reminded that the texts have been truncated to remove the matched pattern and any text following it. For the *text* model (Tables 4.8), many of the important features are numerals from the text. Some of these numerals appear in a unique record in that class (e.g. 'fraction 52.7 %' in borderline, 'volume 79.1 ml' in marked, 'volume 55.2 ml' in gross), and for this reason they are strongly associated with that class. For the *text∪kb* model (Table 4.9), important features include fewer numerals from the text. Instead, the EF attribute from the

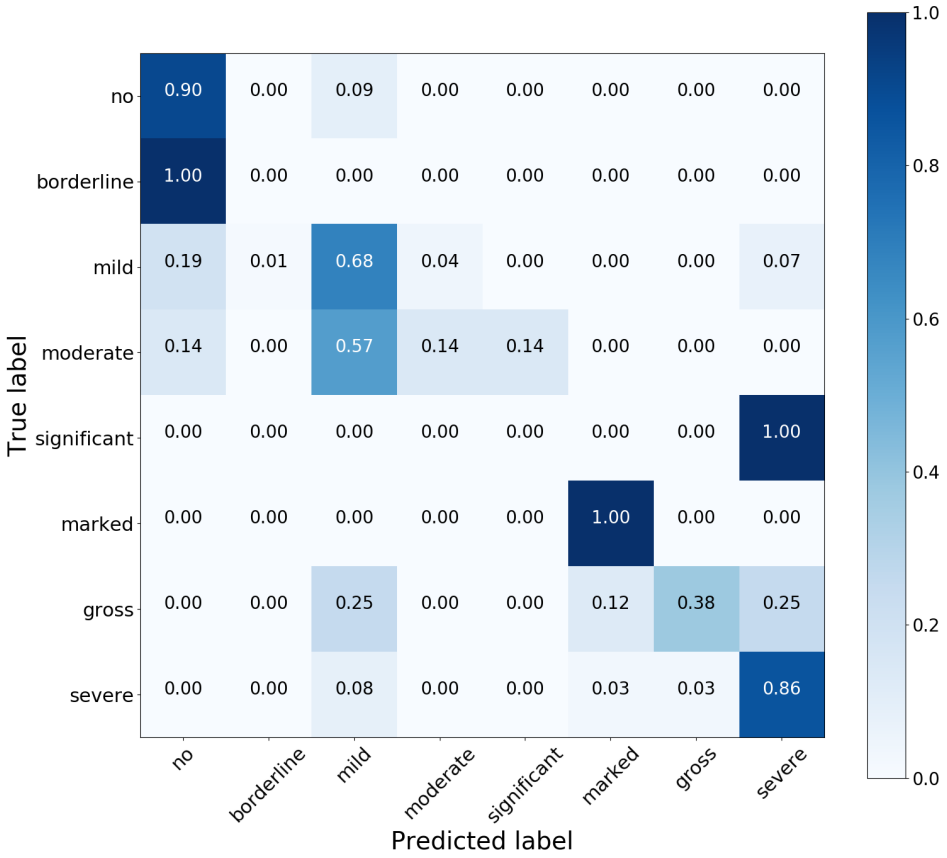

Figure 4.2: Confusion matrix for the *text*∪*kb* model. The counts have been normalised per row.

KB has a large positive weight for the 'no', 'borderline', and 'mild classes, and a large negative weight for the 'marked', 'gross', and 'severe' classes.

Decision Areas Figure 4.3 shows the decision areas of the *kb* model, that is, the combinations of the EF, EDV, age and sex attributes from the KB for which the classifier assigns a different label. Additionally, the plots are overlaid with datapoints from the development set for patients of the same sex and in the same decade of life., along with their true label for the dilation class. For increasing values of EDV or decreasing values of EF, the model changes its decision from 'no', through 'borderline' (or 'mild', 'moderate', or 'significant') to 'gross' (or 'severe' or 'marked'). Age and sex are also important at shaping the decision areas. Most misclassifications (coloured points in areas of different colour) occur between classes with neighbouring decision areas.

Table 4.8: Most important features from the *text* model.

	Positive	Negative
no	normal, volume 110.6, effusions ., hcm, normal lv, history, 57, 65.1, chambers is, trivial mr	37.3, 38.8, % thoracic, 21.5, 35.4, 44.5, dilated with, volume 44.5, 44.5 ml, 23.2
borderline	119.8, volume 103.8, 103.8 ml, 103.8, fraction 52.7, 52.7, 52.7 %, and, and mild, mild tr	ventricular, lv, wall, la, with mild, with, moderate, and tr, % conclusion, ;
mild	37.3, 35.4, fraction 45.1, 45.1 %, 105.8, 105.8 ml, volume 105.8, 45.1, fraction 21.7, 21.7 %	function :, volume 110.6, severe, diameter, resting, mr ., ml lv, 57, tr ., % ed
moderate	ventricular function, fraction 70.3, conclusion, 70.3 %, 70.3, rate 73, 73 bpm, resting ventricular, 73, output 10.2	resting cardiac, cardiac function, ml end, % conclusion, results end, . ., volume :, bpm conclusion, left ventricular, are
significant	42.9 %, fraction 42.9, 42.9, volume 59.5, 59.5 ml, 59.5, rate 53, 53 bpm, is, volume 67.7	conclusion, conclusion thoracic, % conclusion, with, /, arrangement, -, g, the, are
marked	volume 79.1, 79.1 ml, 79.1, volume 76.9, 76.9 ml, 76.9, 59.7 ml, volume 59.7, 59.7, 57.4	44.6, 52.6, 52.6 ml, volume 52.6, % history, : the, of the, in, arrangement of, trivial
gross	23.2, 55.2 ml, volume 55.2, 55.2, 94.2 ml, 94.2, volume 94.2, mr ., : the, 24.2	and mild, mild tr, ;, are mildly, % ed, 51, non, noted ., there is, dilated .
severe	38.8, 44.5 ml, volume 44.5, 44.5, 21.5, severe, 51 bpm, rate 51, dcm, 51	49.9 %, fraction 49.9, be, . bilateral, . trivial, 60.9, 79.1, 79.1 ml, the, volume 79.1

4.1.5 Discussion

Findings The results indicate that numerically information is crucial to the performance of models in a word prediction classification task. Our most important findings are:

- A model that uses a combination of textual and non-textual input features (*text*∪*kb*) can perform better than models that use either only textual (*text*) or only non-textual (*kb*) features in a word prediction classification task. This holds for most results averaged across classes (Table 4.6) and for the results

Table 4.9: Most important features from the *text∪kb* model. Features from the KB in **bold**.

	Positive	Negative
no	**attribute:sex=male**, normal, **attribute:EF**, history, of, of great, history :, : normal, function :, the cardiac	**attribute:sex=female**, mild, aorta, (,), there, 4, : there, aortic, significant
borderline	/, and, **attribute:EF**, 119.8, volume 103.8, 103.8 ml, 103.8, fraction 52.7, 52.7, 52.7 %	wall, :, ml stroke, ventricular, with, vessels ., la, is, **attribute:sex=female**, stroke
mild	was, ml, with, rv, mildly, :, are, with mild, **attribute:EF**, dimension	mm, non dilated, ml lv, rca,) ., aorta, non, 103.8, 103.8 ml, volume 103.8
moderate	ventricular function, resting ventricular, ventricular, on, mild, conclusion history, wall, %, rca, echo	resting cardiac, cardiac function, the, ml end, cardiac, are, : the, dilated, . ., systolic
significant	is, mm, measures, dilated, dilated ., of great, has,) ., . ., ra	:, conclusion, with, conclusion thoracic, : normal, anatomical, normal anatomical, anatomical arrangement, arrangement ., % conclusion
marked	volume 79.1, 79.1, 79.1 ml, arrangement ., anatomical arrangement, normal anatomical, anatomical, volume 59.7, 59.7 ml, 59.7	and, **attribute:EF**, vessels, great, great vessels, heart, in, bpm, rate, heart rate
gross	the, in, : the, mr ., . the, la, 21, mr, are, rv	;, **attribute:EF**, heart, history, ml ejection, of, there is, ventricular function, function, : end
severe	severe, heart, and, right, 60, male, chamber, lv systolic, old male, . there	rv, 21, . resting, **attribute:sex=male**, mildly, cabg, mr ., **attribute:EF**, mildly dilated, /

of several individual classes (Table 4.7).

- Numerical information is important enough that many of the most important features of the *text* model were n-grams that contained numerals (Table 4.8). Such n-grams might be sparse, and their categorical nature might not be the most efficient representation for the underlying continuous attribute. In the important features of the *text∪kb* model, numerals are largely displaced by continuous attributes from the KB (Tables 4.9).

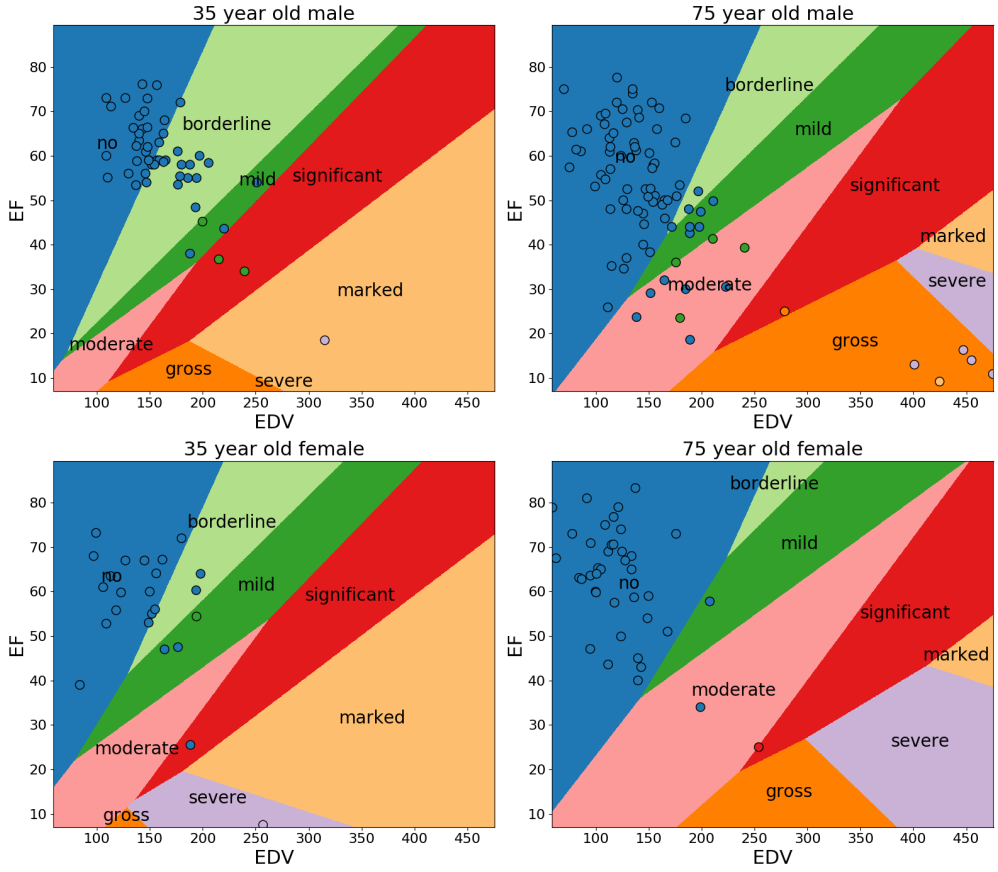

Figure 4.3: Decision areas of the *kb* model. The model classifies instances to different classes of dilation status for different combinations of attributes from the KB (EDV, EF, age, and sex).

- The importance of smoothness of the continuous attributes is seen in the decision areas of the *kb* model (Figure 4.3).

Limitations There are several limitations for this experiment:

- **Limitations from Data** The dataset contained texts from several clinicians, and there is expected to be lexical variation among them. We tried to alleviate this issue by normalising the labels, e.g. conflating adverbs and adjectives. The data came from a single clinical speciality, i.e. cardiology, and some of the classes had few instances. We had to filter out instances with missing values in the KB, which makes our techniques not applicable to datapoints with missing values and might skew the data from their natural distribution; therefore, the findings are valid only for the subset of data with no missing

KB values. Finally, we truncated the data to only keep the text up to (but not including) the matched pattern.

- **Limitations from Models** The models were not deep and did not encode interactions of features. The n-gram features do not encode longer text dependencies and can suffer for sparsity. Finally, all results are to be interpreted as correlations and not as causation, e.g. severe dilation comments on the dimensions of the heart and should depend on the EDV attribute, but most our models pick up a dependence on the EF.

In general, our results and findings should be validated externally.

4.2 Regression: From Words to Numbers

In the previous section, we noticed that numerical attributes from the KB can suffer a high percentage of missing values, namely more than 37% of the EDV and EF attributes in the dataset had missing values (Table 4.4). Several reasons can lead to that, e.g. the value was not measured, not reported, lost, or corrupted. Some of the missing values can be reconstructed from the rest of the data with one of two approaches: i) the value of the attribute can be exactly extracted from the text; ii) the value of the attribute can be approximately predicted from the text or from other attributes in the KB. The former is only possible when the numerical information is explicitly mentioned in the text, e.g. 'the EF was 35%'; for this reason, we are interested to investigate whether the value can be predicted in the absence of such explicit mentions. In this section, we investigate the importance of textual information for predicting the values of numerical attributes in the KB. In particular, we are interested in models that predict values for the EDV and EF attributes.

4.2.1 Approach

We address the problem as a regression task with the following specifications:

- **Input** The input features can be either textual features from the text or non-textual features from the KB;

- **Output** We consider two continuous outputs from the KB: the EDV and the EF.

For both output attributes, we will compare the performance of regression models that use various combinations of textual and non-textual input features.

4.2.2 Data

Preprocessing We follow the same preprocessing steps as in the previous experiment of the previous section, except that we do not truncate the texts to remove the matched patterns and we do not filter out any label classes, which leaves us with $10,024$ instances in total. We simulate the absence of direct numerical mentions of the attributes in the text by additionally replacing all numerals with an 'UNKNUM' token. It should be noted that even though the corresponding numerical ranges of the EF and EDV attributes overlap for the various classes (see next paragraph), the classes themselves are mutually-exclusive, as only one of the corresponding words could appear in the matched pattern. Other than merging the adjective and adverb forms (e.g. 'mild' and 'mildly') into a single class (and merging low frequency matches into the other class), we chose not to merge classes with overlapping numerical ranges, so as to retain the high granularity of the gradable words and to not inject additional clinical knowledge into the models.

Statistical Description Figures 4.5 and 4.4 show the histograms for the distributions of the EF and EDV attributes, respectively, for each class. Similarly to their means and standard deviations (Table 4.5), the histograms for the 'unmatched' class resemble those for the 'no' class. A possible explanation is that for most cases that the pattern failed to match, there is simply no dilation. On the contrary, the histogram of the 'other' class appears different.

4.2.3 Experimental Setup

Models For each target variable, we train the following linear regression models:

1. **mean** This model only has a bias and no other input features (equivalent to predicting with the mean value from the training data).

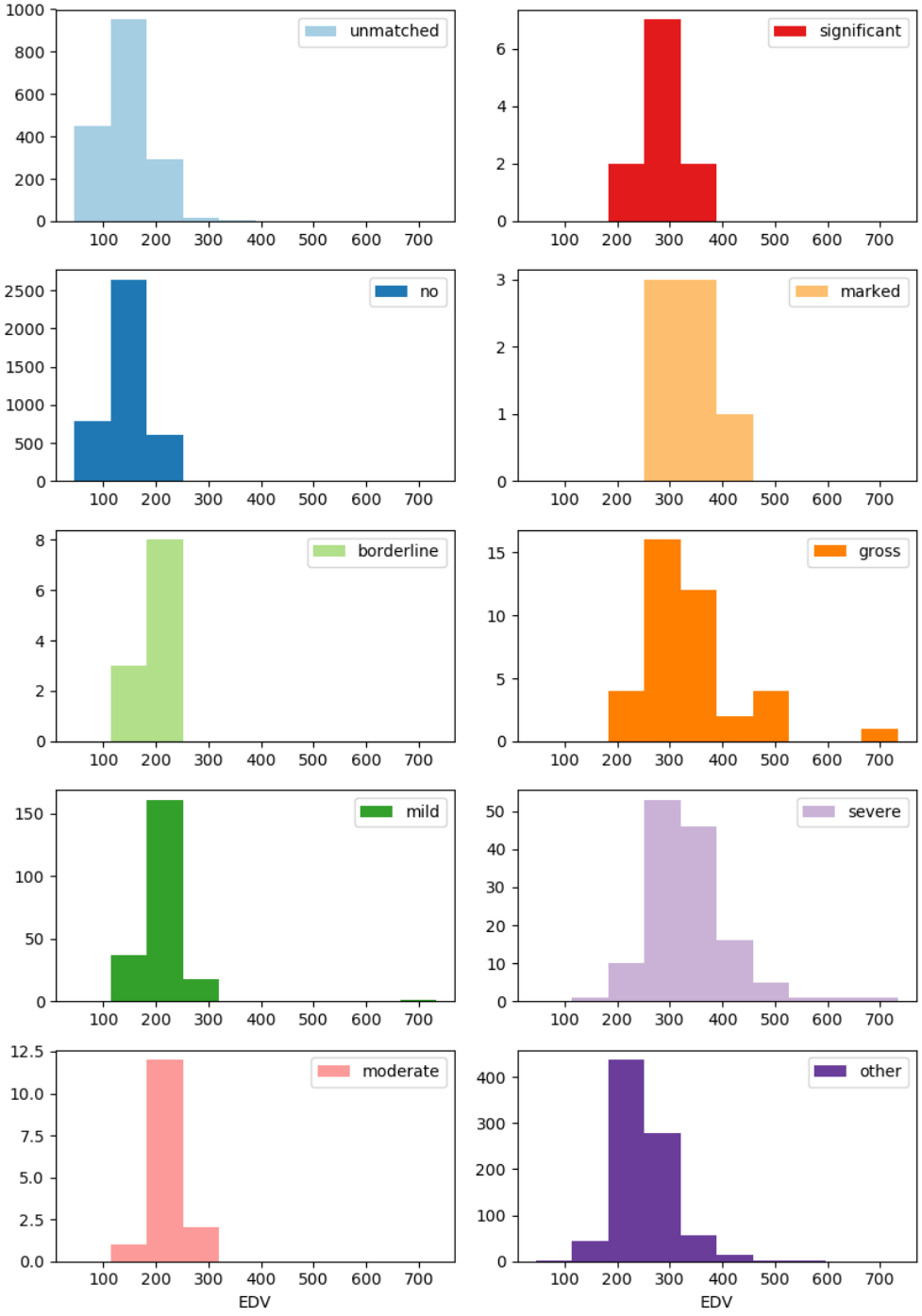

Figure 4.4: Histograms of EDV for class of dilation.

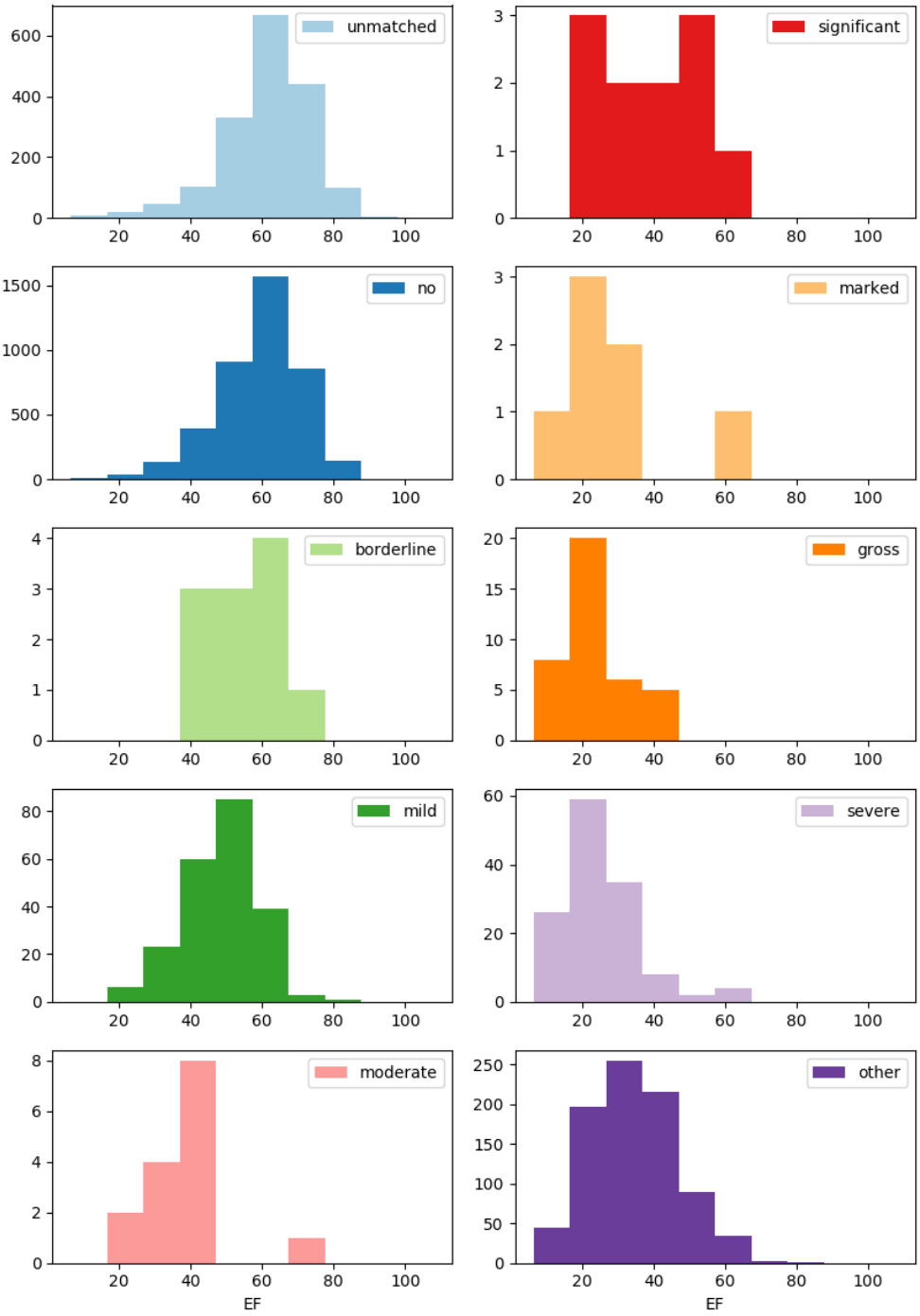

Figure 4.5: Histograms of EF for class of dilation.

Table 4.10: Results for regression for EF and EDV. Root mean square error (RMSE), mean absolute error (MAE), and median absolute error (MdAE) are absolute metrics; mean absolute percentage error (MAPE) is a relative metric. Best model in **bold**.

model	RMSE	MAE	MdAE	MAPE
		EF		
mean	15.44	12.14	9.88	31.91
kb	15.13	11.88	9.59	31.10
label	11.51	8.91	7.41	20.69
text	6.82	5.29	4.39	12.13
all	**6.71**	**5.21**	**4.31**	**11.93**
		EDV		
mean	60.51	44.91	36.05	30.12
kb	57.83	42.02	33.07	27.77
label	41.76	31.85	26.96	21.98
text	39.02	29.17	23.22	19.57
all	**35.93**	**26.40**	**21.47**	**17.68**

2. **kb** This model uses as features the values of attributes from the KB (age and sex).

3. **label** This linear regression model uses as features the label of the dilation class extracted through pattern matching (including the unmatched and other classes). This is equivalent to predicting with the mean value from the training data for that class.

4. **text** This model uses n-gram features from the texts (all 1-grams and 2-grams that appear at least 3 times in the texts of the training data).

5. **all** This model uses the combined features from all previous models.

Training Details All models are trained with coordinate descent to minimise an RMSE objective and regularised with elastic net regularisation (i.e. both L1 and L2), where the hyper-parameters are selected to optimise the training objective on the development set.

Table 4.11: Breakdown of results for regression for EF and EDV The results report root mean square error (RMSE) for each dilation class. Best model for each class in **bold**.

	class									
model	**unmatched**	**no**	**borderline**	**mild**	**moderate**	**significant**	**marked**	**gross**	**severe**	**other**
	EF									
mean	11.26	34.68	19.72	12.97	15.68	11.48	34.21	36.04	23.85	12.85
kb	14.95	34.40	20.94	12.75	15.77	11.38	33.18	40.59	23.16	12.53
label	16.78	**9.05**	**3.26**	10.84	7.83	10.67	10.79	18.47	12.62	11.36
text	**0.37**	13.10	6.89	**5.60**	4.58	5.80	10.41	**11.57**	**6.75**	6.30
all	0.59	13.20	6.34	5.75	**4.55**	**5.65**	**9.97**	12.12	6.88	**6.13**
	EDV									
mean	37.46	135.66	148.27	55.28	76.47	39.54	189.91	75.74	94.49	48.58
kb	28.33	136.72	154.00	48.44	78.39	37.05	185.59	98.91	90.90	45.26
label	24.64	**46.27**	58.54	**26.81**	51.09	34.01	75.38	**5.65**	50.69	44.07
text	37.80	49.63	50.90	35.67	39.01	32.81	79.16	11.10	48.62	35.81
all	**12.43**	49.43	**49.02**	29.25	**35.33**	**28.85**	**71.67**	11.17	**45.25**	**33.01**

4.2.4 Results

Regression Results Table 4.10 shows regression evaluation metrics (RMSE, MAE, MdAE, and MAPE) for the two target variables (EF and EDV) and for the various models. All results are evaluated on the test set. For all metrics and for both targets, the *all* model has the best performance. Furthermore, models that also used textual features (*all*, *text*, and *label*) perform better than models that only used non-textual features (*kb* and *mean*).

Breakdown of Regression Results Table 4.11 shows a breakdown of the RMSE metric within subsets of the test data for each individual class. For all classes and for both targets, the best performance is achieved by one of the models that use textual features (*all*, *text*, or *label*). Models that used textual features (*all*, *text*, or *label*) performed better than models that only used non-textual features (*kb* and *mean*) for most classes and targets, with the exception of the *text* model for EDV and the *label* model for EF in the unmatched class. For the unmatched class, the *all* and *text* achieved particularly low errors. It should be noted that even though we truncated documents during pre-processing to remove all text after matching the

Table 4.12: Most important features from the *text* model.

	Positive	Negative
EF	hyperdynamic, dynamic, dynamic systolic, severely dilated, severe tr, severe ar, good systolic, exertional, severe mr, to severely	with poor, severe lv, severe global, severely, poor systolic, severely impaired, with severe, impairment, impaired, reduced systolic
EDV	severely dilated, : severe, increased lv, severe lv, severely, grossly, : dilated, dilatation, dilation, thinned	- dilated, non dilated, not dilated, small lv, severe tr, hyperdynamic, in systole, female, moderate tr, af

pattern, only for the unmatched class we kept the entire document (since none of the patterns matched). Therefore, the instances of the unmatched class contained more information that the models can use to make their predictions.

Important Features Table 4.12 shows the 10 most important features for each target from the respective *text* model, that is, the features with the most positive and negative coefficients. Important features for the EDV and EF targets correspond to phrases that discuss the ventricular size and systolic function, respectively. For example, the phrase 'severely dilated' (positive feature) correlates with a higher EDV value, and 'non dilated' (negative feature) with a lower EDV value.

4.2.5 Discussion

Findings The results indicate that textual information is crucial to the performance of models in a regression task. Our most important findings are:

- A model that uses a combination of textual and non-textual input features (*all*) can perform better than models that use either only textual (*label* and *text*) or only non-textual (*kb*) features in a regression task. This holds for all results averaged across classes (Table 4.10) and for the results of several individual classes (Table 4.11).

- Models that use textual input features (*all*, *label*, or *text*) can perform better than models that only non-textual (*kb*) features in a regression task. This holds for all results averaged across classes (Table 4.10) and for the results of most individual classes (Table 4.11).

- Important textual features correctly identify common phrases that discuss the respective numerical attributes (Table 4.12).

Limitations There are several limitations for this experiment:

- **Limitations from Data** There were some outliers, particularly for the EDV attribute. Additionally, we had to filter out instances with missing values in the KB, which makes some of our techniques not applicable to datapoints with missing values and might skew the data from their natural distribution (due to non-random missingness); therefore, the findings are valid only for the subset of data with no missing KB values. Finally, we did not treat outliers in any special way, even though they might require different modelling assumptions (e.g. come from a different distribution).

- **Limitations from Models** The models were not deep and did not encode interactions of features. The n-gram features do not encode longer text dependencies and can suffer for sparsity. Finally, all results are to be interpreted as correlations and not as causation.

In general, our results and findings should be validated externally.

Chapter 5

The Effect of Numbers on Language Modelling

'Something's going on, and it has to do with that number. The answer is there.'

— Sean Gullette as 'Max Cohen', in *Pi*

In Chapter 4 we found that numbers can be useful for modelling single words; in this chapter we look to investigate whether this finding holds for longer sequences of words, i.e. texts. We focus on the effect of numerical magnitude of numbers and numerals in language modelling. We also consider the effect on downstream tasks that use language models, namely text prediction (word prediction and completion) and semantic error detection and correction.

Research Question We frame our investigation as the following research question:

Research Question Q.3
Can inputting numbers into a language model improve its ability to model/predict/correct texts (sequences of words and numerals)

Scope Similarly to Chapter 4, we will address this research question in the expert domain of clinical records. Table 5.1 shows an example of a clinical record and highlights the spans of text that depend on numerical information from the KB or the text. Numerical information is represented in the value of the attribute-value pairs from the KB, e.g. *age:76*, and in numeral in the text, e.g. '*76* year old man'.

Table 5.1: Example of text with dependencies on a KB and on numbers. Text in **bold** relays information found in the KB, and underlined text relays information that depends on numerical attributes earlier found in the text or in the KB.

KB	Text
age : 76 sex : male LV EDV (ml) : 378.85 LV ESV (ml) : 279.83 LV SV (ml) : 99 LV EF (%) : 26.1	Left Ventricular Functional Analysis Results **End diastolic volume 378.85 ml End systolic volume 279.83 ml Stroke volume 99.0 ml Ejection fraction 26.1 %** History: **76 year old man. Dilated LV** of unknown aetiology. Unobstructed coronary arteries. Thoracic anatomy: Normal arrangement of cardiac chambers and great vessels. [...] **The left ventricle is severely dilated; systolic function is severely impaired.** [...] Conclusion: **Severe LV dilation and systolic impairment**. [...]

Structure of Chapter This chapter is structured as follows:

- In Section 5.1, we investigate neural language models that depend on numbers (numerical magnitude of numerals in text and numbers in a KB).
- In Section 5.2, we investigate the performance of number-dependent language models in the task of text prediction (word prediction and completion).
- In Section 5.3, we investigate the performance of number-dependent language models in the task of semantic error detection and correction.

This chapter is partly based on work previously published by the author (Spithourakis et al., 2016a,b).

5.1 Language Modelling

Humans regularly compose texts that describe, summarise or are otherwise based on some context. Often this context includes numerical information, as shown in the example from the clinical domain in Table 5.1, where a clinician has composed a clinical report given a KB about a patient. The mechanism of generation of such data has been described in Chapter 4, and more information can be found in the

Appendix. In the example of Table 5.1, the clinician has transcribed numbers from the KB into numerals in the text (e.g. age:*76* to '*76* year old man') or has translated them into textual descriptions after performing expert reasoning (e.g. EF:*21.1* to 'systolic function is *severely impaired*'). In this section, we are interested in developing and evaluating language models that similarly depend on numerical information, namely numerical magnitude of numerals in the text and numbers in the KB.

5.1.1 Approach

Let $s_1, s_2, ..s_T$ be a sequence of tokens that is the text, where s_t is the token at position t. Let e_t^{token} be the token embedding for token s_t that a neural language model uses as input.

Challenges In order to build language models that incorporate numerical magnitude, we need to address the following challenges:

- **Variable data types** The KB contains both numerical and categorical attributes; the text contains tokens that are numerals and others that are not. The property of numerical magnitude is defined only for numbers from the KB (the value of numerical attributes) and numerals from the text.
- **Variable structure** The KB can have attributes with missing values or have different sets of attributes across different instances; likewise, different texts can have a different number of tokens, i.e. different lengths.

Magnitude-Dependent Embeddings To address the variable data type challenge for the text, we define the *magnitude* of token s_t as:

$$m_t = \begin{cases} |\text{number}(s_t)| & \text{if } s_t \text{ is numeral} \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

where number$(.)$ is a conversion function that takes a numeral as input and returns its numerical value. This straightforward definition sets the magnitude of non-numeral words to zero. We proceed to define a *magnitude-dependent embedding*

for token s_t by concatenating the magnitude for that token to its embedding:

$$e_t^{\text{magnitude}} = \begin{bmatrix} e_t^{\text{token}} \\ m_t \end{bmatrix} \tag{5.2}$$

This magnitude-dependent embedding becomes the input to the RNN's recurrence function. We rely on the RNN to address the variable structure challenge, since RNN's can deal with sequences of arbitrary length. Therefore, through our definition of the magnitude-dependent embeddings and the use of a RNN, we have addressed the variable data type and the variable structure challenges, respectively, and we can build magnitude-dependent language models.

Dependence on KB Building a language model that is conditional on the KB requires that we solve the variable data type and the variable structure challenges. We achieve that by reusing and extending our solution for the text data by following the steps below:

1. **KB Lexicalisation** We convert the KB into text, for which we have already addressed the two challenges. This is done by converting the list of *attribute:value* pairs into a delimited string. Table 5.2 shows an example of a lexicalised KB. Lexicalisation of structured data, e.g. knowledge bases, ontologies, etc., is a standard technique in data-to-text language generation (Wiseman et al., 2017; Liang et al., 2009; Reiter and Dale, 2000; Belz, 2008; Lebret et al., 2016).

2. **Encoder-Decoder Framework** A neural network (encoder), e.g. a RNN, is used to build a representation of the lexicalised KB, h_{KB}. This becomes the initial hidden state of the language model's RNN (decoder).

The resulting language model depends on the attributes of the KB and the magnitude of their values.

5.1.2 Data

Our dataset comprises 16,003 anonymised clinical records from the London Chest Hospital. Each patient record consists of a text report and an accompanying struc-

Table 5.2: Example of KB lexicalisation. The KB entry is a collection of attribute:value pairs, while the lexicalised KB entry is text.

KB			Lexicalised KB
age	:	76	"age = 76 ; sex = male ; LV_EDV_ml = 378.85 ; LV_ESV_ml = 279.83 ; LV_SV_ml = 99 ; LV_EF_% = 26.1 ;"
sex	:	male	
LV EDV (ml)	:	378.85	
LV ESV (ml)	:	279.83	
LV SV (ml)	:	99	
LV EF (%)	:	26.1	

tured KB.

Preprocessing We preprocess the dataset following the steps below:

1. **Defining data folds** We randomly split the data into training, development and test sets using a 70/20/10 ratio, respectively.

2. **Tokenisation** We tokenise at the whitespaces and all punctuation (with the exception of the decimal point separator, to avoid splitting numerals).

3. **Lowercasing** We convert all characters to lowercase.

Descriptive Statistics for the Text Table 5.3 summarises descriptive statistics of the dataset and for a vocabulary of the 1,000 most frequent tokens from the training data. Numerals account for a large part of this vocabulary (>40%); at the same time they suffer from high out-of-vocabulary rates (>40%), despite constituting only a small proportion of the total tokens in the data (4.3%).

Structure and Content of the Knowledge Base The KB contains a list of *attribute:value* pairs for each patient. The attributes describe demographic (age and sex) and clinical (e.g. end diastolic and systolic volumes for the left and right ventricles as measured through magnetic resonance imaging) information about the patient. This information was available to the clinician at the time of the compilation of the text. In total, there are 20 possible attributes, of which one is categorical (sex) and the rest are numerical (age, EF, EDV, etc.). The KB contains, on average, 5 *attribute:value* pairs per patient.

Table 5.3: Statistics for clinical dataset. Counts for non-numeral (*words*) and *numeral* tokens reported as percentage of counts for *all* tokens. Out-of-vocabulary (OOV) rates are for the vocabulary of 1,000 most frequent words in the train data.

		train	**dev**	**test**
	#documents	11,158	1,625	3,220
#tokens/doc	all	204.9	204.4	202.2
	words	95.7%	95.7%	95.7%
	numeral	4.3%	4.3%	4.3%
OOV rate	all	5.0%	5.1%	5.2%
	words	3.4%	3.5%	3.5%
	numeral	40.4%	40.8%	41.8%

5.1.3 Experimental Setup

Models We define the following neural LMs:

1. **baseLM** This model is a LM built on a single layer Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997). The model uses input and output token embeddings for the vocabulary of $V = 1000$ most frequent tokens from the training set. The vocabulary also includes two unknown tokens, for OOV numerals and other OOV words, respectively. The dimensions for all model parameters (internal matrices, input and output embeddings) are set to $D = 50$.

2. **kb** This is a LM that is conditional on the lexicalised KB. The encoder and decoder have the same architecture (similar to the *baseLM* model) and their weights are tied. The lexicalised attributes and delimiters are added to the input vocabulary.

3. **num** This model is a LM that uses magnitude-dependent token embeddings for the input tokens and is otherwise similar to the *baseLM* model.

4. **kb∪num** This is a LM that is conditional on the lexicalised KB. It uses magnitude-dependent token embeddings for the input tokens and is otherwise similar to the *kb* model.

Table 5.4: Results for language modelling on the test set. We report perplexity (PP) and adjusted perplexity (APP), evaluated on different subsets of tokens (all tokens, only words, and only numerals). Best results in **bold**.

	all		words		numerals	
model	**PP**	**APP**	**PP**	**APP**	**PP**	**APP**
baseLM	14.96	22.11	13.93	17.94	72.38	2289.47
kb	14.52	21.47	13.49	17.38	74.48	2355.77
num	9.91	14.66	9.28	11.96	42.67	1349.59
kb∪num	**9.39**	**13.88**	**8.80**	**11.33**	**39.84**	**1260.28**

Training Details All models are trained to minimise a cross-entropy loss, with 20 epochs of back-propagation and gradient descent with adaptive learning rates (AdaDelta) (Zeiler, 2012) and mini-batch size set to 64. The hyper-parameters were selected based on a small search on the development set around values commonly used in the literature.

5.1.4 Results

Perplexities Table 5.4 shows the perplexity and adjusted perplexity evaluation metrics for the various models. All results are evaluated on the test set for different subsets of its tokens (all tokens, only words, and only numerals). The subsets have different OOV-rates, thus perplexities are not comparable between subsets. Adjusted perplexity is not sensitive to OOV-rates; thus, it allows for meaningful comparisons between subsets. For all subsets and both metrics, the best performance is achieved by models that use magnitude-dependent embeddings (*kb*∪*num*, followed by *num*). For the subset of words, models that are conditional on the lexicalised KB (*kb* and *kb*∪*num*) perform slightly better than unconditional models (*baseLM* and *num*, respectively). For the subset of numerals, the adjusted perplexity of all models is much higher (2 orders of magnitude) than for the subset of words. The disparity between perplexity and adjusted perplexity is the greatest for the subset of numerals.

Word Probabilities Figure 5.1 shows the ratios of word probabilities between the best and baseline models, i.e. *kb*∪*num* and *baseLM*, respectively, at various positions in texts from the development set. The ratio at each position shows how many

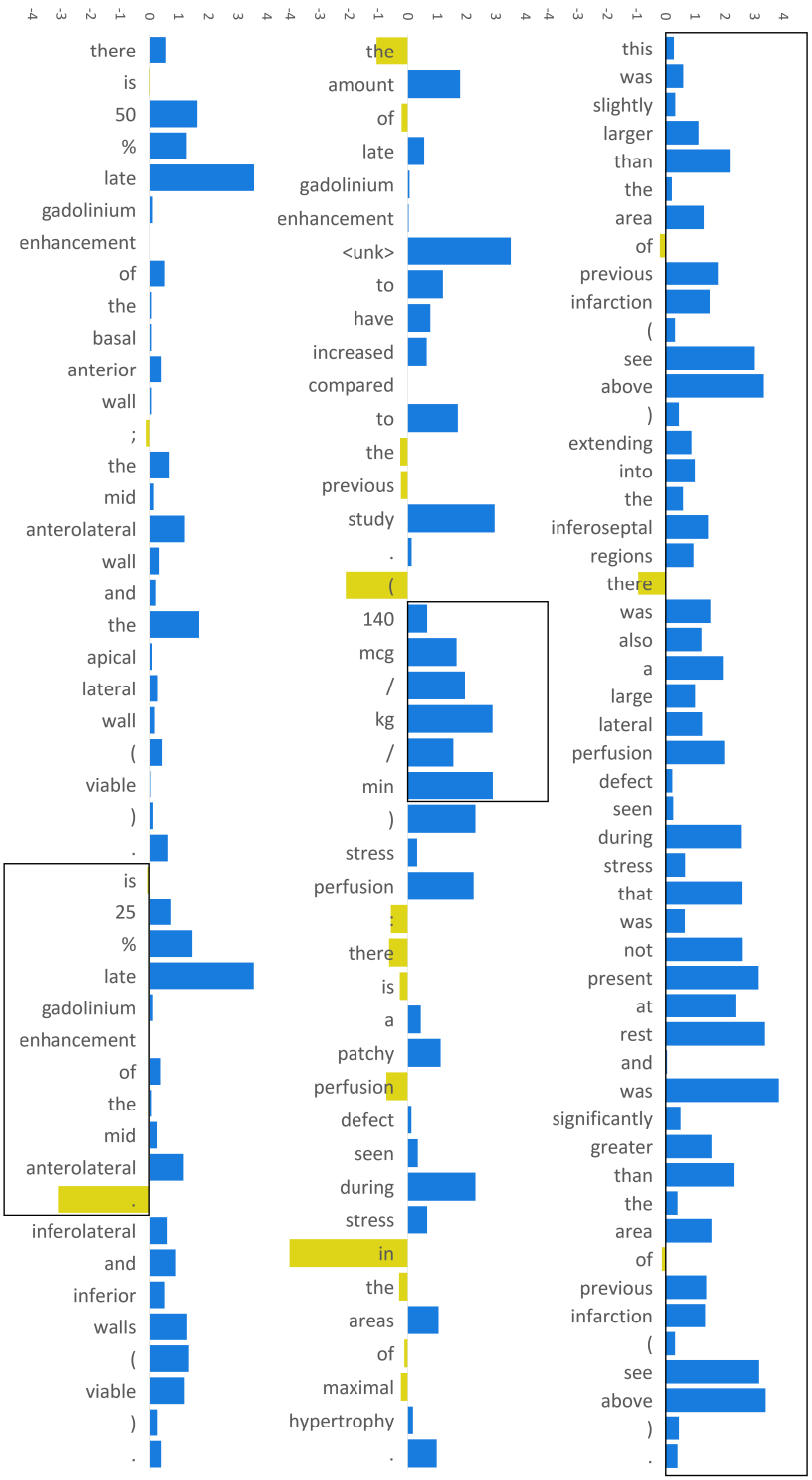

Figure 5.1: Ratios of word probabilities under different language models. The ratio is from the *kb*∪*num* to the *baseLM* model. The text snippets are from the development set. Boxes highlight areas with interesting phenomena.

more times the probability under *kb*∪*num* is greater than (or less than) the probability under *baseLM* for that word, i.e. if the ratio at a position is r, the probability for that word appearing in that position estimated using the *kb*∪*num* model is $|r|$ times higher (or lower, if r is negative) than the same probability estimated using the *baseLM* model. For most positions, the ratio is positive, i.e. *kb*∪*num* returns higher probabilities than *baseLM* for most tokens, including some numerals (e.g. '25% late gadolinium enhancement', '140 mcg/kg/min'), which is in agreement with the models' perplexity scores (see Table 5.4). However, the ratio turns negative at certain positions with erroneous choices of words, e.g. missing words ('there' and 'wall') and early sentence termination ('.') in the snippet '[there] is 25% late gadolinium enhancement of the mid anterolateral [wall]'; this indicates that the proposed model might be useful for detecting errors, which will be explored in Section 5.3.

5.1.5 Discussion

Findings The results indicate that numerical magnitude and conditioning on a KB can improve the perplexity of language models. Our most important findings are:

- Primarily, using magnitude-dependent embeddings can greatly improve language modelling performance, as models that use magnitude-dependent embeddings (*kb*∪*num* and *num*) perform much better than the other models (*kb* and *baseLM*, respectively; Table 5.4).

- Secondarily, conditioning on a lexicalised KB can slightly improve language modelling performance, as models that condition on the KB (*kb*∪*num* and *kb*) perform slightly better than the other models (*num* and *baseLM*; Table 5.4). This is possibly because many texts begin by recounting the information from the KB in a format that resembles the lexicalised KB, as in the example of Table 5.1.

- The improvements from using magnitude-dependent embeddings and from conditioning on a lexicalised KB are orthogonal, as model with the best performance (*kb*∪*num*; Table 5.4) implemented both.

- Numerals are harder targets than other words for language models, as all models had a high adjusted perplexity for that subset (Table 5.4). This finding can be easily overlooked if one only examines the unadjusted perplexity, which does not account for the OOV-rate, or the perplexities evaluated on all tokens, which is overwhelmed by the high proportion of non-numeral words (Table 5.3).

Limitations There are several limitations for this experiment:

- **Limitations from Data** Because of out tokenisation, all numerals represented non-negative numbers. This seems to be a minor limitation, as all values in the KB were non-negative.

- **Limitations from Models** In a straightforward manner, we have defined the magnitude of non-numeral words to be zero, even though it is technically undefined. Another possible problem with this representation is that the input can become too large, causing numerical instabilities (e.g. overflows) at training or test time.

5.2 Application: Text Prediction

In this section, we investigate whether our findings about the importance of numerical magnitude translate into downstream language modelling tasks. In particular, we experiment with applications of text prediction, where a system assists a user in text data entry tasks. The input and output of the system are:

- **Input** The user actively types a stream of characters into a system;

- **Output** The system present to the user a ranked list of suggestions for each next word.

Two variants of text prediction are word prediction and text prediction:

1. **Word Prediction** The list of suggestions is static and is not updated until the user moves on to the next word. Word prediction can facilitate exploration of vocabulary or promotion of standardised vocabularies.

Table 5.5: Task examples for word prediction and completion. The input is the sequence of characters that have been entered so far, and the output is a list of suggestions for predicting/completing the next word. The context is often relevant to the quality of the output.

Task	input	output
Word Prediction	76 year old	male gentleman man patient
	Dilation of the lv is	severe gross significant mild
Word Completion	76 year old ma	le n
	Dilation of the lv is s	evere ignificant
	context	
	KB: age:76 sex:male EDV:378.85 EF:26.1	

2. **Word Completion** The list of suggestions is interactively updated as the user enters more characters. The user can choose to auto-complete the word with the first element in the list, typically by typing a special character (e.g. tab). The main goal of a word completion is to reduce the time and cost of text data entry.

Table 5.5 shows examples for the two tasks with systems that demonstrate the desirable behaviour. The systems prioritise the words 'male' and 'severe', because the KB contains the attribute:value pairs *sex:male* and *EDV:378.85*, respectively.

Text Prediction in the Clinical Domain Many clinical data entry systems provide text prediction features to assist clinicians in the compilation of clinical texts, such as discharge summaries (Chen et al., 2012), brain MRI reports (Cannataro et al., 2012) and radiology reports (Eng and Eisner, 2004). Text prediction can lead to

Algorithm 1 Word completion

Input: $\mathcal{V}$ is set of vocabulary words, *scorer* returns score for word in current position
Output: next word to be written

function COMPLETEWORD($\mathcal{V}$, *scorer*)
  $prefix \leftarrow$ ''
  $lexicon \leftarrow \mathcal{V}$
  loop
    $lexicon \leftarrow$ {tokens in *lexicon* starting with $prefix$}
    $best \leftarrow \underset{token \in lexicon}{\text{argmax}} \; scorer(token)$
    Display *best*
    $char \leftarrow$ read next char
    if $char = TAB$ **then**
      return best ▷ Auto-complete
    else if $char = WHITESPACE$ **then**
      return $prefix$ ▷ Next word
    else
      $prefix \leftarrow prefix + char$ ▷ Append
    end if
  end loop
end function

keystroke savings and time reductions for data entry of texts (Gong et al., 2016; Hua et al., 2014). Finally, text prediction has been used to standardise clinical texts (Sirel, 2012; Lin et al., 2014) and to improve adherence to fields (Gong et al., 2016) and response accuracy (Hua et al., 2014).

5.2.1 Approach

Word Prediction We use a LM to estimate the probability of the next word given the history of typed words and any additional context. Regarding the list of suggestions, the system presents the N-best list of words with the highest probability.

Word Completion We use a similar approach to word prediction, but we additionally use a prefix matching algorithm to interactively remove suggestions that do not match the characters already typed by the user. Algorithm 1 describes our approach that is based on interactive prefix matching against the lexicon. The algorithm takes as input the set of known vocabulary words and a scoring function that returns the goodness of a word in the current position and context, which again can be the word

probability from a language model. Initialisation sets the prefix to an empty string and the lexicon to the whole vocabulary (lines 2-3). Iteratively, words that do not match with the prefix are removed from the lexicon (line 5), the best word from the lexicon according to the scorer is found and displayed to the user (lines 6-7) and the user can respond with a key (line 8). If the user inputs the special character, the best word is automatically completed (lines 9-10). If the user inputs a whitespace character, the algorithm terminates (11-12). This is the case when no matching word is found in the vocabulary. If any other character is typed, it is appended to the prefix and another iteration begins.

5.2.2 Data

Simulated Data Entry An automated evaluation for both tasks can be executed by simulating a user who types the text character by character. The character stream can come from a dataset of finalised texts. For both tasks, we assume that the word from the dataset is the correct word. For the word completion task, we assume that the user types the special key to auto-complete the word as soon as the correct suggestion becomes available at the top of the suggestion list.

5.2.3 Experimental Setup

Models Our systems for word prediction and completion are based on the following language models:

- **baseLM, kb, num, and kb∪num** These language models are defined and trained as per Section 5.1.

For the word completion task, we additionally consider the following oracle systems:

- **theoretical** This oracle system returns a list of suggestions that contains only the correct word.
- **vocabulary** This oracle system returns a list of suggestions that contains only the correct word (similarly to the *theoretical* oracle), if the correct word is in the known vocabulary; otherwise, it returns an empty list of suggestions.

Table 5.6: Results for word prediction on the test set. Recall at different ranks (R@rank) and mean reciprocal rank (MRR). At rank=1, recall is equal to precision (i.e. R@1=P@1). Best system performance in **bold**.

Word Prediction						
	model	**R@1/P@1**	**R@3**	**R@5**	**R@10**	**MRR**
system	baseLM	8.36	18.38	25.03	36.66	17.19
	kb	45.27	59.97	65.18	71.18	54.49
	num	21.13	35.45	43.66	53.72	31.91
	kb∪num	**51.76**	**66.36**	**71.28**	**77.10**	**60.71**

Table 5.7: Results for word completion on the test set. Keystroke savings (KS) are equivalent with recall (R). Unnecessary distractions (UD) is inversely related to precision (P). Best system values in **bold**.

Word Completion					
	model	**KS/R**	**UD**	**P**	**F1**
oracle	theoretical	58.87	0.00	100.0	74.11
	vocabulary	54.48	0.00	100.0	70.54
system	baseLM	34.35	6.17	13.96	19.85
	kb	43.17	3.11	24.34	31.13
	num	39.31	4.38	18.60	25.25
	kb∪num	**44.81**	**2.76**	**26.60**	**33.38**

5.2.4 Results

Word Prediction Table 5.6 shows evaluation metrics for the word prediction task (recall at various ranks and mean reciprocal rank) for the various models. All results are evaluated on the test set. At rank 1, recall is equivalent to precision. Relative model performance is consistent across all metrics. The best performance is achieved by models that are conditional on the lexicalised KB (*kb*∪*num*, followed by *kb*). Models that use magnitude-dependent embeddings (*kb*∪*num* and *num*) perform slightly better than models that are not aware of magnitudes (*kb* and *baseLM*, respectively). The *baseLM* model fared the worst, even more than what we would expect based on its similar perplexity scores to the *kb* model (Table 5.4); this observation indicates that, at each token, the probabilities assigned by the *baseLM* and *kb* models to the ground-truth next word are comparable, but the ranking of the lists of suggestions from the two models are very different. This is possible, because

tiny changes in the probabilities are sufficient to cause perturbations in the rankings of the lists. Word prediction with the *baseLM* model might suffer from low recall, e.g. at rank 10 its recall is 36.66%, i.e. only 36.66% of all suggestion lists of length 10 contain the correct word; other models can have improved performance at lower ranks (equivalently, with shorter lists of suggestions), e.g. at rank 5 recall is 43.66% for *num*, 65.18% for *kb*, and 71.28% for *kb*∪*num*.

Word Completion Table 5.7 shows evaluation metrics for the word completion task (keystroke savings, unnecessary distractions, precision, recall, and F1 score) for the various models and for the oracles (theoretical and vocabulary). All results are evaluated on the test set. Keystroke savings is equivalent to recall, and unnecessary distractions are inversely related to precision. Relative model performance is consistent across all metrics. The best performance is achieved by models that are conditional on the lexicalised KB (*kb*∪*num*, followed by *kb*). Models that use magnitude-dependent embeddings (*kb*∪*num* and *num*) perform slightly better than models that are not aware of magnitudes (*kb* and *baseLM*, respectively). Word completion with the *baseLM* model can already lead to 34.35% keystroke savings, i.e. the user needs to press 34.35% fewer characters to type the same text. Other models can lead to further savings (39.31% for *num*, 43.17% for *kb*, and 44.81% for *kb*∪*num*), while reducing the unnecessary distractions (average number of characters the user has to scan through before choosing to auto-complete the word). The performance of any system is bounded from above by that of the *vocabulary* and *theoretical* oracles (Trnka and McCoy, 2008) at 54.48% and 58.87% keystroke savings, respectively. The oracles have perfect precision and zero unnecessary distractions, because they never make any wrong suggestions.

Exploring the List of Suggestions Table 5.8 shows the list of suggestions (top 5 suggestions and ranks for other interesting suggestions) by the various models for word prediction with a text from the development set. In that text, the next token corresponds to the dilation status class, similarly to Chapter 4. It happens that the *baseLM* makes the correct suggestion with its first item in the list (in agreement with the ground-truth, 'non'). The correct suggestion appears further down in the list for

Table 5.8: Word prediction for sample document from the development set. Top-5 suggestion lists for the next word (ground-truth is '**non**' and ranks for interesting terms from the complete lists of different systems.

Text: left ventricular function analysis results end diastolic volume 131.49 ml end systolic volume 79.22 ml stroke volume 52.3 ml ejection fraction 39.8 % [...] lv systolic function is moderately impaired . non dilated atria . non dilated rv [...] lv is [**non** dilated ...]

system:	**baseLM**	**kb**	**num**	**kb∪num**
rank	**suggestions**			
1	**non**	normal	normal	preserved
2	normal	preserved	dilated	normal
3	dilated	**non**	not	dilated
4	preserved	good	preserved	not
5	not	mild	**non**	with
suggestion	**ranks**			
non-dilated	10	11	8	13
dilated	3	8	2	3
non	1	3	5	7
moderately	41	33	37	36
mildly	6	6	7	6
severely	29	23	28	27

the other models; still, their top suggestions (i.e. 'normal' and 'preserved') seem reasonable, and would not be inconsistent with an occurrence of the pattern 'the lv is non dilated' later in the text. An inconsistent suggestion that indicates the presence of dilation (e.g. 'dilated') does appear higher than the correct suggestion ('non') for the *num* and *kb∪num* models; however, such continuations are grammatically correct and, at least, semantically relevant, and appear towards the top of the list also for *baseLM*. Finally, no list clearly separates words that might indicate presence (e.g. 'dilated', 'mildly', etc.) or absence (e.g. 'non', 'not', 'non-dilated', etc.) of dilation.

Sensitivity to Numerical Magnitude For the text in Table 5.8, we replace the numerical information (numerals in the text and lexicalised KB) with various possible configurations for the numerals that should favour a different word (e.g. 'mildly' or 'severely') to appear in the selected position. We inspect the list of suggestions

by the various models for various substitute numerals that are unseen in the training data (unknown numerals). The rankings at the top of the list remain mostly unchanged, and we do not notice any difference at the ranks shown in Table 5.8. The insensitivity of the rankings to permutations in the numbers in the text and KB suggests that the models might under-utilise the magnitude component of the magnitude-dependent embeddings at test time. It is possible that the number-aware model forgets the numbers from the KB because of the text that is later encounters.

5.2.5 Discussion

Findings The results indicate that numerical magnitude and conditioning on a KB can improve the performance of language models in tasks of text prediction (word prediction and word completion). Our most important findings are:

- Primarily, conditioning on a lexicalised KB can greatly improve the performance of language models in text prediction tasks, as models that condition on the KB (*kb*∪*num* and *kb*) perform much better than the other models (*kb* and *baseLM*, respectively) in word prediction (Table 5.6) and word completion (Table 5.7).

- Secondarily, using magnitude-dependent embeddings can slightly improve the performance of language models in text prediction tasks, as models that use magnitude-dependent embeddings (*kb*∪*num* and *num*) perform slightly better than the other models (*kb* and *baseLM*, respectively) in word prediction (Table 5.6) and word completion (Table 5.7). The insensitivity of the rankings in Table 5.8 to permutations in the numbers in the text and KB suggests that the models might under-utilise the magnitude component of the magnitude-dependent embeddings at test time.

- The improvements from conditioning on a lexicalised KB and from using magnitude-dependent embeddings are orthogonal, as model with the best performance (*kb*∪*num*; Tables 5.6 and 5.7) implemented both.

- Based on the evaluation for the simulated user, a real user would experience poor quality of service with text prediction systems that use the lan-

guage model that neither conditions on a lexicalised KB nor uses magnitude-dependent embeddings (*baseLM*). The quality of service would be greatly improved with systems that use a similar language model that implements both techniques (*kb*∪*num*). Namely, the user would benefit from higher recall even with shorter suggestion lists (from 36.66% with 10 suggestions to 71.28% with 5 suggestions; Table 5.6), when exploring the vocabulary with word prediction, and higher keystroke savings (from 34.35% to 44.81%) with even fewer unnecessary distractions (from 6.17 to 2.76 distracting characters per word; Table 5.7), when entering text with word completion.

Limitations Additionally to the limitations from Section 5.1, there are additional limitations for this experiment:

- **Limitations from Evaluation** The system suggestions might influence the text produced by the user. We ignore any possible effects of the feedback loop between the user and the system, and assume that the each next correct word is the word found in the text data. Higher keystroke savings and lower unnecessary distractions might in practice not translate to time and effort savings, because of unaccounted factors (e.g. system interface, human behaviour, etc.).

5.3 Application: Semantic Error Detection and Correction

In this section, we investigate whether our findings about the importance of numerical magnitude translate into downstream language modelling tasks. In particular, we experiment with applications of semantic error detection and correction, where a system assists a user at detecting and correcting semantic errors in texts. The input and output of the systems are:

- **Input** The user enters a text into the system that needs to be checked for semantic errors. A semantic error is an inconsistency or contradiction between statements in the text or between the text and the attribute-value pairs of a

Table 5.9: Task examples for semantic error detection and correction. The input is a text, and the output is a prediction for error detection/correction. The context is often relevant to the quality of the output.

Task	input	output
Error Detection	76 year old male	detect=FALSE
	Dilation of the lv is severe	detect=FALSE
	76 year old female	detect=TRUE
	Dilation of the lv is mild	detect=TRUE
Error Correction	76 year old male	correct={}
	Dilation of the lv is severe	correct={}
	76 year old female	correct={female → male }
	Dilation of the lv is mild	correct={mild → severe }
	context	
	KB: age:76 sex:male EDV:378.85 EF:26.1	

linked KB. These inconsistencies may stem from oversight, lack of reporting guidelines or negligence.

- **Output** The system issues an alert to the user, if it system detects a semantic error; otherwise, the system notifies the user that no semantic error was detected (in many applications, this notification is suppressed, to minimise nuisance to the user).

For each task, the alert to the user contains the following information:

1. **Semantic Error Detection** The alert notifies the user that a semantic error has been detected in the text.

2. **Semantic Error Correction** The alert notifies the user that a semantic error has been detected in the text and recommends a text edit to correct the error.

Table 5.9 shows examples for the two tasks with systems that demonstrate the desirable behaviour. The error detection system is triggered to detect errors by the words ‘female’ and ‘mild’, because they are inconsistent with the attribute:value pairs *sex:male* and *EDV:378.85* from the KB, and the error correction system proposes to correct them to the more consistent words ‘male’ and ‘severe’, respectively.

Semantic Errors in the Clinical Domain In a review of clinical records, 60% of texts about patients had errors (contradictions, copying errors, documentation errors, etc.) with an average of 7.8 errors per patient (Weir et al., 2003). Some of these errors, e.g. prescription errors can be potentially fatal (Lisby et al., 2005). Overall, it is estimated that medical errors account for 44,000 to 98,000 deaths per year in the United States alone (Donaldson et al., 2000). The adverse effects from errors in clinical texts can be reduced by prospective audits of errors (Macaulay et al., 1996) or by preventive guidelines for text composition, e.g. write 'more than' or 'less than' in prescriptions instead of '$<$' or '$>$' which can be confused (Institute for Safe Medication Practices, 2017).

5.3.1 Approach

Error Detection Our approach to error detection hinges on our approach to error correction: our error detection system is triggered (i.e. detects an error) when our error correction system is triggered (i.e. suggests a correction).

Error Correction A statistical model chooses the most likely correction from a set of possible correction choices. If the model scores a corrected hypothesis higher than the original text, the correction is accepted. For a selected original text H_0, our approach to error correction follows these steps:

1. **Generate candidates** Use a hypothesis generator function, G, to generate a set of candidate corrected texts $G(H_0) = \{H_1, ..., H_M\}$. A simple hypothesis generator uses confusion sets of semantically related words to produce all possible substitutions.

2. **Score candidates** Use a scorer model, s, to assign a score $s(H_i) \in \mathbb{R}$ to each hypothesis H_i. The scorer is based on

3. **Make recommendation** Propose a correction if an alternative hypothesis scores higher that the original hypothesis. Semantic error detection can be derived by saying that an error is detected when the Error Detection and Correction (EDC) system would recommend any hypothesis different than the original text.

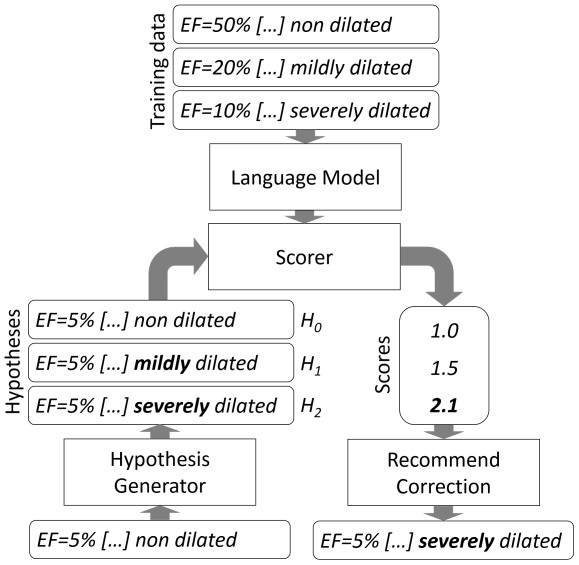

Figure 5.2: Semantic error correction using language models.

Our approach to EDC is outlined in Figure 5.2. We use a hypothesis generator that proposes lexical substitutions based on confusion sets extracted from a development set, which would make the approach a weakly supervised approach. For the scorer model, we use a likelihood ratio test between the original text (null hypothesis, H_0) and each candidate correction (alternative hypotheses, H_i), i.e. $s(H_i) = \frac{p(H_i)}{p(H_0)}$. The assigned score represents how much more probable a correction is than the original text. The probability of observing a text, $p(H_i)$, is estimated using LMs.

5.3.2 Data

Constructing a Dataset with Errors Ideally, one should evaluate the tasks of semantic error detection and correction on a dataset that contains real errors that have been annotated by humans. If such a resource is not available, one can corrupt an existing, trusted dataset by infusing simulated error. Such errors can come from common word substitutions observed in a development set. To evaluate EDC, we generate a 'corrupted' dataset of semantic errors from the test part of the 'trusted' dataset (Table 5.3, last column), by following the following steps:

1. We manually build confusion sets (Table 5.10) by searching the development set for words related to numeric quantities and grouping them if they appear in similar contexts.

Table 5.10: Confusion sets for errors in the clinical data. The confusion sets were identified from the clinical data.

description	confusion set
intensifiers (adv):	*non*, *mildly*, *severely*
intensifiers (adj):	*mild*, *moderate*, *severe*
units:	*cm*, *mm*, *ml*, *kg*, *bpm*
viability:	*viable*, *non-viable*
quartiles:	*25*, *50*, *75*, *100*
inequalities:	$<$, $>$

2. Then, for each text in the trusted test set we generate an erroneous text by sampling a substitution from the confusion sets. Documents with no possible substitution are excluded.

The resulting 'corrupted' dataset is balanced, containing 2,926 correct and 2,926 incorrect texts.

5.3.3 Experimental Setup

Models We use an oracle hypothesis generator that has access to the ground-truth confusion sets (Table 5.10). For error detection, we binarise the output of the correction models by detecting an error when the correction model recommends a correction. For error correction, we consider systems with scorers based on the following language models:

- **baseLM, kb, num, and kb∪num** These language models have exactly the same architecture and trained parameters as those in Section 5.1.

Additionally, we consider systems with following random baseline scorers:

- **always** This scorer randomly assigns scores from a uniform distribution to the corrected texts and the lowest score to the original text, i.e. this system *always* proposes a (random) correction;

- **random** This scorer randomly assigns scores from a uniform distribution to each text (including corrected and original), i.e. this system *randomly* proposes to correct or not;

Table 5.11: Results for semantic error detection on the test set. Precision (P), Recall (R), and F-scores (F1 and F0.5). Best results in **bold**.

Error Detection				
model	**P**	**R**	**F1**	**F0.5**
always	50.00	100.0	66.67	55.56
random	50.27	90.29	64.58	55.16
baseLM	57.51	94.05	71.38	62.36
kb	56.86	94.43	70.98	61.78
num	58.87	94.70	72.61	63.69
kb∪num	**60.48**	**95.25**	**73.98**	**65.24**

5.3.4 Results

Semantic Error Detection Table 5.11 shows evaluation metrics for the semantic error detection task (precision, recall, and F scores) for the various models. All results are evaluated on the test set. For all metrics, the best performance is achieved by models that use magnitude-dependent embeddings (*kb*∪*num*, followed by *num*). Error detection with the *baseLM* model can already lead to 94.05% recall and 57.51% precision, i.e. a user could use the system to detect 94.05% of all errors in the texts, but only 57.51% of the triggered detections would be desired (the rest are nuisance alerts). Precision and recall are summarised by the F0.5 metric, which is 62.36% for the *baseLM* model; for the other models, the F0.5 performance might be slightly better (63.69% for *num* and 65.24% for *kb*∪*num*) or worse (61.78% for *kb*). Mainly for the recall metric, the random baselines (*always* and *random*) set a high standard: the *always* baseline is triggered by all texts, and so it detects all errors (perfect recall), but half of its alerts are nuisance (50% precision); the *random* baseline is triggered more often for texts with more alternative hypotheses (i.e. more items in the confusion set) and again triggers many nuisance alerts.

Semantic Error Correction Table 5.12 shows evaluation metrics for the semantic error correction task (precision, recall, F scores, and mean average precision) for the various models. All results are evaluated on the test set. Relative model performance is consistent across all metrics. For all metrics, the best performance is achieved by models that use magnitude-dependent embeddings (*kb*∪*num*, followed

Table 5.12: Results for semantic error correction on the test set. Precision (P), Recall (R) and F-scores (F1 and F0.5). Best results in **bold**.

Error Correction				
model	**P**	**R**	**F1**	**F0.5**
always	6.13	12.26	8.18	9.20
random	5.73	10.29	7.36	8.13
baseLM	39.54	64.66	49.07	53.36
kb	37.46	62.20	46.76	50.98
num	44.25	71.19	54.58	59.18
kb∪num	**45.36**	**71.43**	**55.48**	**59.95**

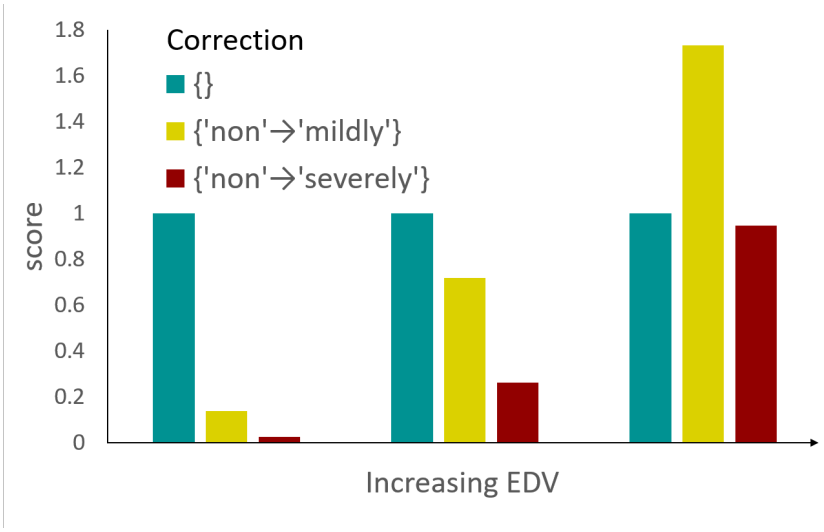

Figure 5.3: Sensitivity of semantic error correction to numbers. Scores of the *num* model for different hypotheses (original: 'non'; alternatives: 'mildly' and 'severely') and for different numerical configurations in a text from the development set (same text as in Table 5.8).

by *num*). Error correction with the *baseLM* model can already lead to 64.66% recall and 39.54% precision, i.e. a user could use the system to correct 64.66% of all errors in the texts, but only 39.54% of the proposed corrections would be desired (the rest are nuisance). Precision and recall are summarised by the F0.5 metric, which is 53.36% for the *baseLM* model; for the other models, the F0.5 performance might be slightly better (59.18% for *num* and 59.95% for *kb*∪*num*) or worse (50.98% for *kb*). For all metrics, all systems perform much better than the random baselines (*always* and *random*).

Sensitivity to Numerical Magnitude Figure 5.3 shows the scores of the *num* model for different hypotheses (original: 'non'; alternatives: 'mildly' and 'severely') and for different numerical configurations in a text from the development set (same text as in Table 5.8). We replace numerals in the text to simulate increasing EDV values (substitute numerals were unseen in the training data). We observe higher scores for 'mildly' and 'severely' as we increase the EDV; however the value exceeds the range of the attribute before 'severely' achieves the highest score.

5.3.5 Discussion

Findings The results indicate that numerical magnitude and conditioning on a KB can improve the performance of language models in tasks of semantic error detection and correction. Our most important findings are:

- Using magnitude-dependent embeddings can improve the performance of language models in semantic EDC tasks, as models that use magnitude-dependent embeddings (*kb*∪*num* and *num*) perform better than the other models (*kb* and *baseLM*, respectively) in semantic error detection (Table 5.11) and correction (Table 5.12). The magnitude-dependent embeddings allowed some sensitivity of the correction outputs to numbers (Figure 5.3).

- Conditioning on a lexicalised KB has mixed results for the performance of language models in semantic EDC tasks, as some models that condition on the KB perform better (*kb*∪*num*) and others worse (*kb*) when compared to unconditional models (*kb* and *baseLM*, respectively) in semantic error detection (Table 5.11) and correction (Table 5.12).

- Still, the best performance was achieved by a model that simultaneously uses magnitude-dependent embeddings and conditions on a lexicalised KB (*kb*∪*num*) in both EDC tasks (Tables 5.6 and 5.7).

- A user of the semantic EDC systems could experience better quality of service with the *kb*∪*num* system than with the *baseLM* system. Namely, the user would benefit from higher recall (from 94.05% to 95.25% of semantic errors)

with fewer nuisance alerts (precision from 57.51% to 60.48%; Table 5.11) for semantic error detection and higher recall (from 64.66% to 71.43% of needed corrections) with fewer mistakes (precision from 29.54% to 45.36%; Table 5.12) for semantic error correction.

Limitations Additionally to the limitations from Section 5.1, there are additional limitations for this experiment:

- **Limitations from Evaluation** We used an oracle hypothesis generator that had access to the ground truth confusion set. Also, dedicated models to address each of the two tasks might be more suitable in practice, We assumed that the texts already were correct and had no other errors.

Chapter 6

Modelling Numerals and Predicting Numbers with Language Models

' "The Answer to the Great Question... Of Life, the Universe and Everything... Is... Forty-two," said Deep Thought, with infinite majesty and calm.'

— Douglas Adams,
The Hitchhiker's Guide to the Galaxy

In Chapter 5 we found the performance of language models can be much worse for numerals than for words. This chapter further investigates this phenomenon and additionally proposes models to improve the performance of language models on numerical information.

Research Questions In this chapter, we will investigate further how we can predict numerical information from words. Specifically, we will focus on the ability of language models to model numerals and predict numbers in the text. We frame this investigation as the following research questions:

Research Question Q.4
How can we improve the ability of language models to model numerals?

Research Question Q.5
How can we improve the ability of language models to predict numbers?

Scope We will address the two research questions in the technical domains of scientific papers and clinical records.

Structure of Chapter This chapter is structured as follows:

- In Section 6.1, we propose models for various strategies for modelling numerals, and we investigate the research question Q.4.
- In Section 6.2, we investigate the research question Q.5.

This chapter is partly based on work previously published by the author (Spithourakis and Riedel, 2018).

6.1 Modelling Numerals with Language Models

Challenges There are two main factors that might limit the performance of existing language models on numerical information:

1. **UNK token and OOV numerals** Numbers that are not in the vocabulary are all mapped to the same token, e.g. 'UNK', which fails to represent numerals with different magnitudes.
2. **Categorical Softmax** The softmax used in the output layer of many existing language models is suitable for categorical variables, but not for the numbers that underlie the numerals, which might be continuous.

Figure 6.1 shows an example of the two challenges. Firstly, many of the numerals are OOV and are mapped to the same 'UNK' token, which results in a uniform probability for all OOV numerals. Secondly, the number that underlies the numeral often stems from a continuous probability density function, which is not captured by the categorical softmax output of existing language models.

6.1.1 Approach

In this section we describe models with different strategies for generating numerals and propose the use of number-specific evaluation metrics that adjust for the high out-of-vocabulary rate of numerals and account for numerical values. We draw inspiration from theories of numerical cognition. The triple code theory (Dehaene

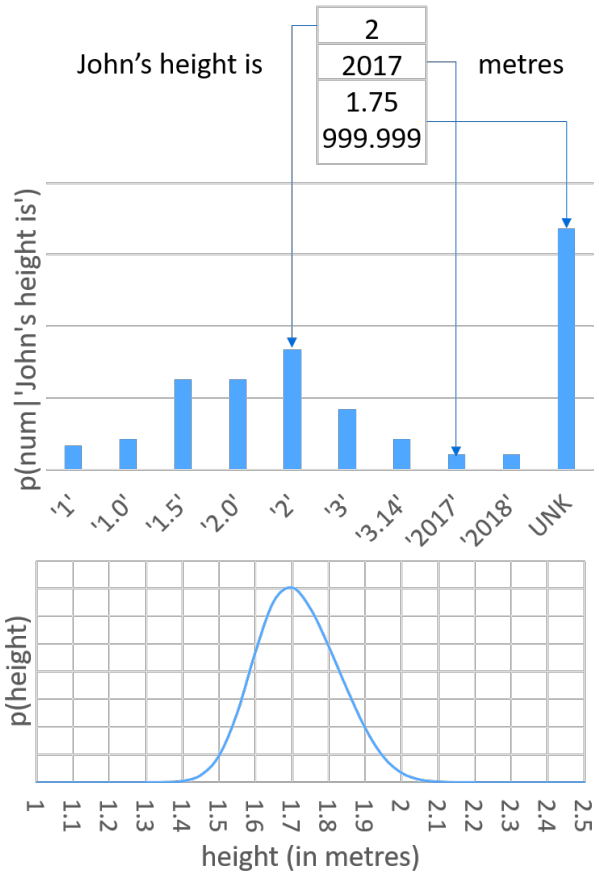

Figure 6.1: Challenges for modelling numerals with language models. Most language models assume a categorical distribution over a fixed vocabulary, which maps all out-of-vocabulary numerals to the same type, e.g. UNK. Furthermore, the categorical distribution does not reflect the smoothness of the underlying continuous distribution of certain attributes.

et al., 2003) postulates that humans process quantities through two exact systems (verbal and visual) and one approximate number system that semantically represents a number on a mental number line. Tzelgov et al. (2015) identify two classes of numbers: i) primitives, which are holistically retrieved from long-term memory; and ii) non-primitives, which are generated online. An in-depth review of numerical and mathematical cognition can be found in Kadosh and Dowker (2015) and Campbell (2005).

Categorical Softmax This class of models assumes that numerals come from a finite vocabulary that can be memorised and retrieved later. The *softmax* model treats all tokens (words and numerals) alike and directly uses Equation 2.18 with

score function:

$$\psi(s_t) = h_t^T e_{s_t}^{\text{token}} = h_t^T E_{\text{out}} w_{s_t}, \tag{6.1}$$

where $E_{out} \in \mathbb{R}^{D \times |\mathcal{V}|}$ is an output embeddings matrix. The summation in Equation 2.18 is over the complete target vocabulary, which requires mapping any out-of-vocabulary tokens to special symbols, e.g. 'UNK' and 'UNKNUM'.

Categorical Softmax with Digit-Based Embeddings The *d-soft* variant considers the internal syntax of a numeral's digits by adjusting the score function:

$$\begin{aligned} \psi(s_t) &= h_t^T e_{s_t}^{\text{token}} + h_t^T e_{s_t}^{\text{chars}} \\ &= h_t^T E_{\text{out}} w_{s_t} + h_t^T E_{\text{out}}^{\text{RNN}} w_{s_t}, \end{aligned} \tag{6.2}$$

where the columns of $E_{\text{out}}^{\text{RNN}}$ are composed of character-based embeddings (see Section 2.1.2) using a character set that comprises: digits (0-9), the decimal point, and an end-of-sequence character. We trivially set $e_{s_t}^{\text{chars}}$ to zero for tokens not covered by this character set. The model still requires normalisation over the whole vocabulary, and the special unknown tokens (i.e. 'UNK' and 'UNKNUM') are still needed.

Hierarchical Softmax A hierarchical softmax (Morin and Bengio, 2005a) can help us decouple the modelling of numerals from that of words. The probability of the next token s_t is decomposed to that of its class c_t and the probability of the exact token from within the class:

$$\begin{aligned} p(s_t|h_t) &= \sum_{c_t \in C} p(c_t|h_t) p(s_t|c_t, h_t) \\ p(c_t|h_t) &= \sigma\left(h_t^T b\right) \end{aligned} \tag{6.3}$$

where the valid token classes are $C = \{\text{word, numeral}\}$, σ is the sigmoid function and b is a D-dimensional vector. Each of the two branches of $p(s_t|c_t, h_t)$ can now be modelled by independently normalised distributions. The hierarchical variants (*h-soft* and *hd-soft*) use two independent softmax distributions for words and numerals. The hierarchical approach allows us to use any well normalised distribution to model each of its branches. In the next subsections, we examine different strategies

for modelling the branch of numerals, i.e. $p(s_t|c_t = \text{numeral}, h_t)$. For simplicity, we will abbreviate this to $p(s)$.

Compositional Strategy Let $d_1, d_2 ... d_N$ be the digits of numeral s. A digit-by-digit composition strategy estimates the probability of the numeral from the probabilities of its digits:

$$p(s) = p(d_1)p(d_2|d_1)...p(d_N|d_{<N}) \tag{6.4}$$

The compositional model (*d-RNN*) feeds the hidden state h_t of the token-level RNN into a character-level RNN (Graves, 2013; Sutskever et al., 2011) to estimate this probability. This essentially is a model that learns a numeral system. This strategy can accommodate an open vocabulary, i.e. it eliminates the need for an "UNKNUM" symbol, as the probability is normalised one digit at a time over the much smaller vocabulary of digits (digits 0-9, decimal separator, and end-of-sequence).

Continuous Density Strategy We propose a model that factorises the probability of a numeral s as a joint probability:

$$p(s) = p(v, r) = p(r)\widetilde{Q}(v|r), \tag{6.5}$$

where v is a random variable for the magnitude of the numeral at precision r, and r is a random variable for the level of discretisation, i.e. how many decimal digits to keep. The following paragraphs discuss how the components of this joint probability can be computed.

Continuous Density Strategy: Mixture of Gaussians We would like to model the number with a probability density function (pdf), $q(v)$; however, the difficulty is that for any continuous random variable, the probability that it equals a specific value is always zero. To resolve this, we consider a probability mass function (pmf) that discretely approximates the pdf:

$$\widetilde{Q}(v|r) = \int_{v-\varepsilon_r}^{v+\varepsilon_r} q(u)du = F(v+\varepsilon_r) - F(v-\varepsilon_r), \tag{6.6}$$

where $F(.)$ is the cumulative density function of $q(.)$, and $\varepsilon_r = 0.5 \times 10^{-r}$ is the number's precision. We use a mixture of Gaussians for the underlying pdf, inspired by the approximate number system and the mental number line (Dehaene et al., 2003):

$$\begin{aligned} q(v) &= \sum_{k=1}^{K} \pi_k \mathcal{N}_k(v; \mu_k, \sigma_k^2) \\ \pi_k &= \text{softmax}\left(B^T h_t\right), \end{aligned} \tag{6.7}$$

where K is the number of components, π_k are mixture weights that depend on hidden state h_t of the token-level RNN, $\mathcal{N}_k$ is the pdf of a normal distribution parametrised by mean $\mu_k \in \mathbb{R}$ and variance $\sigma_k^2 \in \mathbb{R}$, and $B \in \mathbb{R}^{D \times K}$ is a matrix.

Continuous Density Strategy: Discretisation The level of discretisation by converting the numeral into a pattern and use a RNN to estimate the probability of that pattern sequence:

$$p(r) = p(\text{SOS INT\_PART . } \overbrace{\backslash\text{d ... } \backslash\text{d}}^{r \text{ decimal digits}} \text{ EOS}) \tag{6.8}$$

Figure 6.2 summarises the continuous density strategy with a mixture of Gaussians (*MoG*).

Combination of Strategies Different mechanisms might be better for predicting numerals in different contexts. We propose a *combination* model that can select among different strategies for modelling numerals:

$$\begin{aligned} p(s) &= \sum_{\forall m \in M} \alpha_m p(s|m) \\ \alpha_m &= \text{softmax}\left(A^T h_t\right), \end{aligned} \tag{6.9}$$

where M=$\{$h-soft, d-RNN, MoG$\}$, and $A \in \mathbb{R}^{D \times |M|}$. Since both d-RNN and MoG are open-vocabulary models, the unknown numeral token can now be removed from the vocabulary of h-soft.

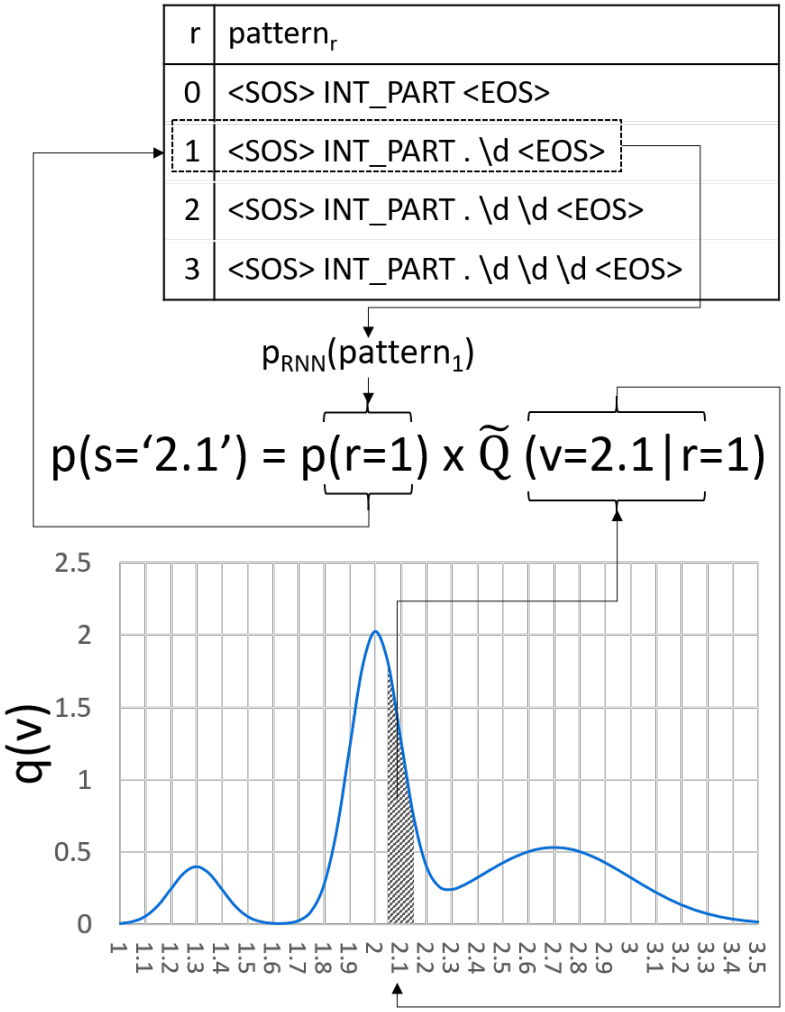

Figure 6.2: Mixture of Gaussians model. The probability of a numeral is decomposed into the probability of its decimal precision and the probability that an underlying number will produce the numeral when rounded at the given precision.

6.1.2 Data

Clinical Data We use the clincial dataset, similary to the previous chapters.

Scientific Data This dataset comprises paragraphs from Cornell's ARXIV [1] repository of scientific articles, with more than half a million converted papers in 37 scientific sub-fields. We used the preprocessed ARXMLIV (Stamerjohanns et al., 2010; Stamerjohanns and Kohlhase, 2008)[2] version, where papers have been converted from LATEX into a custom XML format using the LATEXML [3] tool. We

[1] ARXIV.ORG. Cornell University Library at http://arxiv.org/, visited December 2016
[2] ARXMLIV. Project home page at http://arxmliv.kwarc.info/, visited December 2016
[3] LATEXML. http://dlmf.nist.gov, visited December 2016

Table 6.1: Statistical description of the clinical and scientific datasets: Number of instances, maximum and average lengths, proportions of words and numerals, descriptive statistics of numbers.

	Clinical			Scientific		
	Train	**Dev**	**Test**	**Train**	**Dev**	**Test**
#inst	11170	1625	3220	14694	2037	4231
maxLen	667	594	666	2419	1925	1782
avgLen	210.1	209.1	206.9	210.1	215.9	212.1
%word	95.7	95.7	95.7	96.1	96.1	96.0
%nums	4.3	4.3	4.3	3.9	3.9	4.0
min	0.0	0.0	0.0	0.0	0.0	0.0
median	59.5	59.0	60.0	5.0	4.0	4.5
mean	300.6	147.7	464.8	$\sim 10^{21}$	$\sim 10^{7}$	$\sim 10^{7}$
max	$\sim 10^{7}$	$\sim 10^{5}$	$\sim 10^{7}$	$\sim 10^{26}$	$\sim 10^{11}$	$\sim 10^{11}$

then kept all paragraphs with at least one reference to a table and a number.

Preprocessing For both datasets, we lowercase tokens and normalise numerals by omitting the thousands separator (”2,000” becomes ”2000”) and leading zeros (”007” becomes ”7”). Special mathematical symbols are tokenised separately, e.g. negation (“-1” as “-”, “1”), fractions (“3/4” as “3”, “/”, “4”), etc.

Statistical Description Table 6.1 shows descriptive statistics for both datasets. All numbers are non-negative, because of the tokenisation.

6.1.3 Experimental Setup

Models We consider the following language models:

1. **soft** A model that uses the categorical softmax.
2. **d-soft** A model that uses the categorical softmax strategy with digit-based embeddings.
3. **h-soft** A model that uses the hierarchical softmax strategy.
4. **hd-soft** A model that uses the hierarchical softmax strategy with digit-based embeddings.
5. **d-RNN** A model that uses the digit-by-digit compositional strategy.

6. **MoG** A model that uses the continuous density strategy with a mixture of Gaussians.

7. **combination** A model than uses a combination of *h-soft*, *d-RNN*, and *MoG* strategies.

We set the vocabularies to the 1,000 and 5,000 most frequent token types for the clinical and scientific datasets, respectively. We use gated token-character embeddings (Miyamoto and Cho, 2016) for the input of numerals, and we use token embeddings for the input and output of words, since the scope of our paper is numeracy. We set the models' hidden dimensions to $D = 50$. All our RNNs are LSTMs (Hochreiter and Schmidhuber, 1997).

Training Details We initialise all token embeddings to pretrained GloVe (Pennington et al., 2014) and the biases of the LSTM forget gates to 1.0 (Józefowicz et al., 2015). We train using mini-batch gradient decent with the Adam optimiser (Kingma and Ba, 2014) and regularise with early stopping and 0.1 dropout rate (Srivastava, 2013) in the input and output of the token-based RNN.

Training Details: Means and Variances for MoG For the mixture of Gaussians, we select the mean and variances to summarise the data at different granularities by fitting 7 separate mixture of Gaussian models on all numbers, each with twice as many components as the previous, for a total of $2^{7+1} - 1 = 256$ components. These models are initialised at percentile points from the data and trained with the expectation-minimisation algorithm. The means and variances are then fixed and not updated when we train the language model.

6.1.4 Results

Adjusted Perplexities Table 6.2 shows the adjusted perplexity evaluation metric for the various models. All results are evaluated on the test set for each dataset (clinical and scientific) and for different subsets of tokens (all tokens, only words, and only numerals). All comparisons of adjusted perplexities between subsets, datasets, and models are meaningful. For all tokens of both datasets, the *combination* model achieves the best performance, i.e. it performs better than its individual compo-

Table 6.2: Results for language modelling (adjusted). Adjusted perplexities evaluated on the test set for the clinical and scientific data.

	Adjusted Perplexities					
	Clinical			**Scientific**		
model	**all**	**words**	**numerals**	**all**	**words**	**numerals**
soft	8.91	5.99	58443.72	80.62	51.83	3505856.25
d-soft	8.77	5.91	56164.81	79.47	51.20	3300688.50
h-soft	6.05	4.96	495.95	54.80	49.81	550.98
hd-soft	6.09	4.99	490.14	53.73	48.83	542.70
d-RNN	5.88	**4.95**	263.22	53.70	48.89	**519.80**
MoG	5.88	4.99	226.46	54.37	48.97	683.16
combination	**5.82**	4.96	**197.59**	**53.03**	**48.25**	520.95

Table 6.3: Results for language modelling (unadjusted). Perplexities (unadjusted) evaluated on the test set for the clinical and scientific data. Only comparisons between values with the same exponent (*a*, *b*, etc.) are valid. Best performance for group of comparable results in **bold**. Performances in positions corresponding to best performance in Table 6.2 in *italics*.

	Perplexities					
	Clinical			**Scientific**		
model	**all**	**words**	**numerals**	**all**	**words**	**numerals**
soft	4.28^{a}	4.08^{c}	12.04^{d}	35.79^{f}	33.96^{h}	127.12^{i}
d-soft	4.21^{a}	4.03^{c}	$\mathbf{11.57}^{d}$	$\mathbf{35.28}^{f}$	$\mathbf{33.54}^{h}$	$\mathbf{119.68}^{i}$
h-soft	$\mathbf{4.19}^{a}$	4.00^{c}	11.78^{d}	36.51^{f}	34.73^{h}	122.67^{i}
hd-soft	4.22^{a}	4.03^{c}	11.65^{d}	35.80^{f}	34.04^{h}	120.83^{i}
d-RNN	4.79^{b}	$\boldsymbol{3.99}^{c}$	263.22^{e}	37.98^{g}	34.08^{h}	$\boldsymbol{519.80}^{j}$
MoG	4.79^{b}	4.03^{c}	226.46^{e}	38.45^{g}	34.14^{h}	683.16^{j}
combination	$\boldsymbol{4.74}^{b}$	4.01^{c}	$\boldsymbol{197.59}^{e}$	$\boldsymbol{37.50}^{g}$	$\mathit{33.64}^{h}$	520.95^{j}

nents (*MoG*, *d-RNN*, *hd-soft*); the non-hierarchical softmax variants (*soft* and *d-soft*) have the worst performance. For the subset of numerals, the *combination* model achieves the best performance in the clinical data (followed by *MoG*), and the second best performance in the scientific data (only slightly behind *d-RNN*); the non-hierarchical softmax variants (*soft* and *d-soft*) have by far the worst performance in both datasets (worse than then next best model by factors of 100 and 10,000 for the clinical and scientific data, respectively). For both datasets and all models, performance on numerals is worse than on other models at least by a factor

of 10.

Perplexities (Unadjusted) Table 6.2 shows the (unadjusted) perplexity evaluation metric for the various models. All results are evaluated on the test set for each dataset (clinical and scientific) and for different subsets of tokens (all tokens, only words, and only numerals). Comparisons of results are not meaningful between evaluations that have different OOV-rates, i.e. between different datasets, between different subsets, or between a model with closed (all softmax variants) and a model with open vocabulary of numerals (*d-RNN*, *MoG*, and *combination*). For most permitted comparisons, perplexity follows the same performance patterns as adjusted perplexity (Table 6.2); however, there is not much disparity between the perplexity of non-hierarchical (*soft* and *d-soft*) and hierarchical softmax variants (*h-soft* and *hd-soft*).

Numeral Embeddings: Softmax versus Hierarchical Softmax Figure 6.3 visualises the cosine similarities between the output token embeddings for all numerals in the vocabulary, for the *soft* and *h-soft* models trained on the clinical data. For the non-hierarchical softmax (*soft*), the similarities between all numerals (including the unknown numeral) are positive, whereas, for the hierarchical softmax (*h-soft*), many similarities between numerals (including the unknown numeral) are negative. For both models, similarities are in general higher between numerals with numerically closer magnitudes (along the diagonal in Figure 6.3), between some clusters of numerals (e.g. '1' through '30' and '2010', '2011', '2012', etc.), and between some isolated numerals (e.g. '25', '50', '75', and '100').

Digit Embeddings Figure 6.4 visualises the cosine similarities between the output embeddings for digits from the *d-RNN* model trained on the scientific data. Positive similarities occur only between the decimal point ('.') and 'EOS' symbol and between digits '1' through '9'. For the latter, similarities are in general higher between consecutive digits. Digits that represent smaller integers (e.g. '1', '2') are similar to fewer other digits. The digit '0' is not similar to any other symbol.

Distributions of Significant Digits Figure 6.5 shows the model distribution (from *d-RNN* model estimates, averaged on the development set), empirical distribution

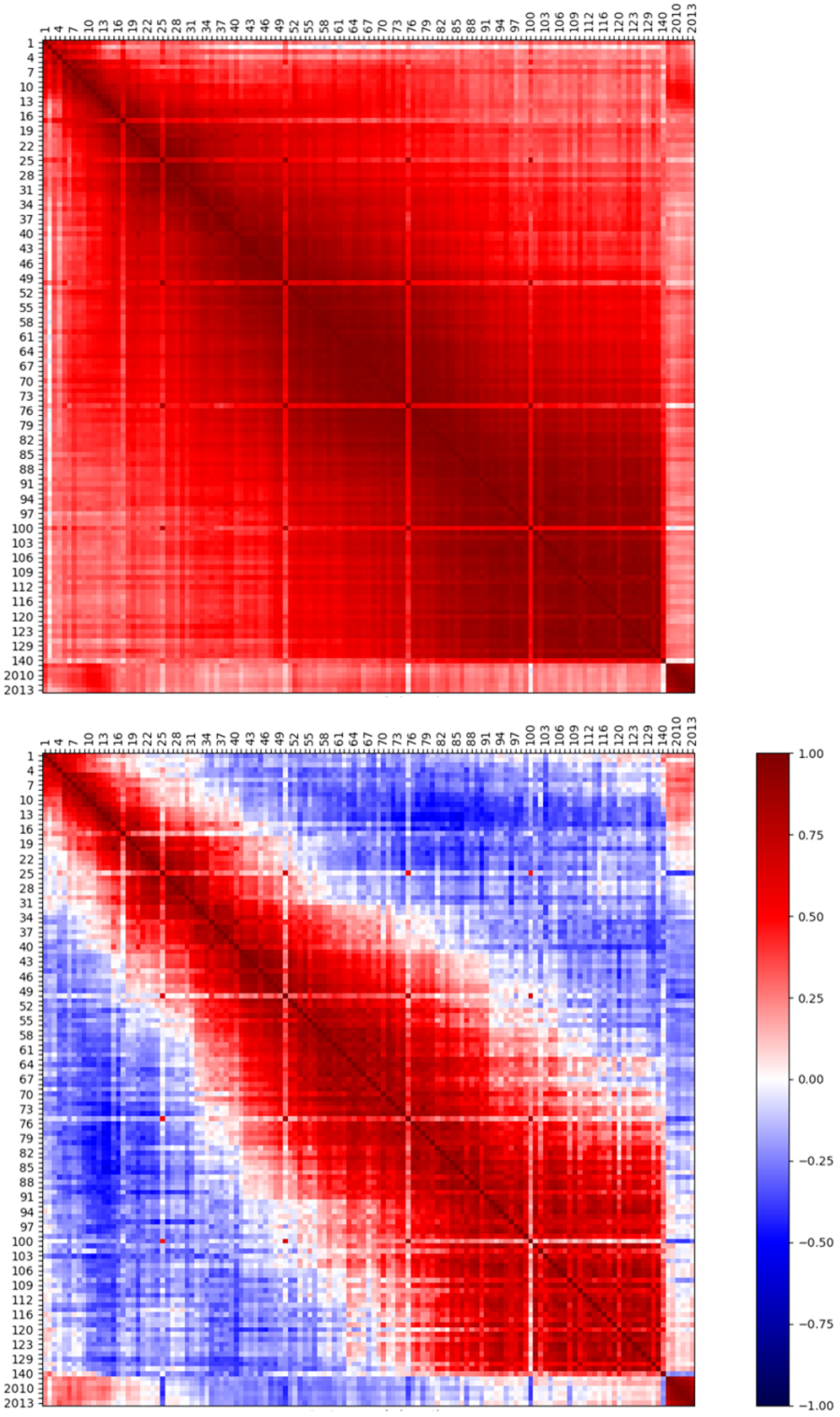

Figure 6.3: Cosine similarities of embeddings for numerals. Embeddings are from the *soft* (top) and *h-soft* (bottom) models trained on the clinical data. Numerals are sorted by magnitude.

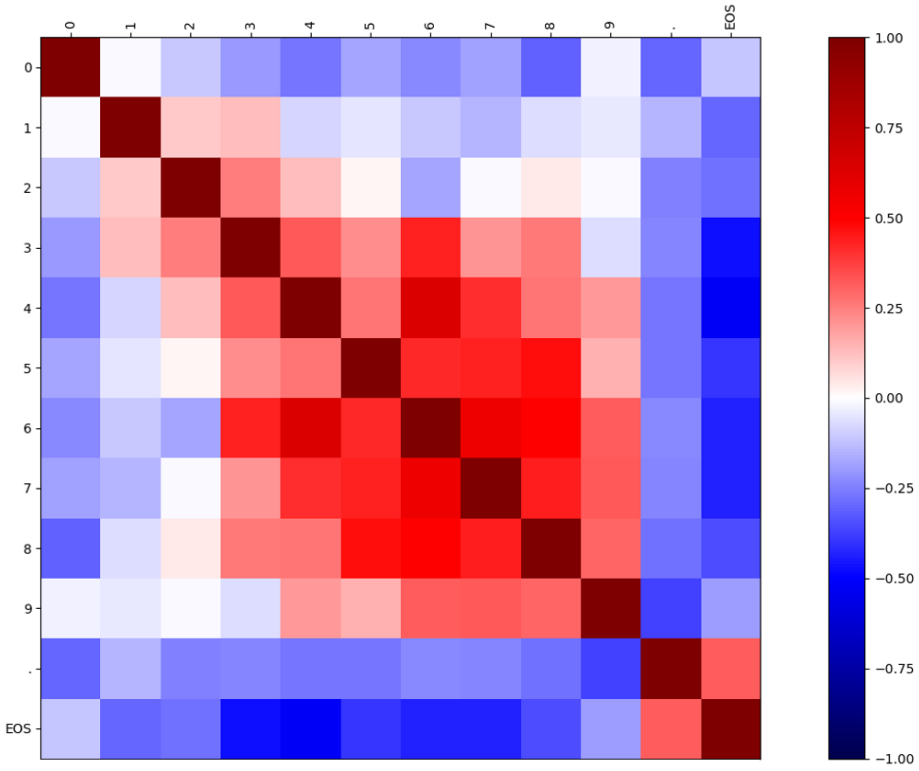

Figure 6.4: Cosine similarities of embeddings for digits from the output layer of the *d-RNN* model trained on the scientific data.

(from training data), and theoretically expected distribution (from Benford's law) for digits at various positions of the numerals in both datasets. *Benford's law* (Benford, 1938), also known as the first-digit law, applies to numerals from many real-life datasets, and predicts that the first digit of a numeral will be '1' with higher probability (about 30%) than '9' ($< 5\%$). Benford's law weakens towards uniformity for later (less significant) digits. The model and empirical distributions are similar to one another; their estimates are occasionally lower (e.g. 1st digit being '3' for clinical data) or higher than theoretically expected (e.g. 1st digit being '5' for clinical and 4th digit being '0' for scientific data), but in general they tend towards uniformity for less significant digits, as theoretically expected.

Combination Model: Strategy Selection Table 6.4 shows the numerals with the highest strategy selection probability for each of strategy of the *combination* model (*h-soft*, *d-RNN*, and *MoG*) and for each dataset (clinical and scientific); it also shows snippets of texts where these numerals appear. The strategy selection probabilities

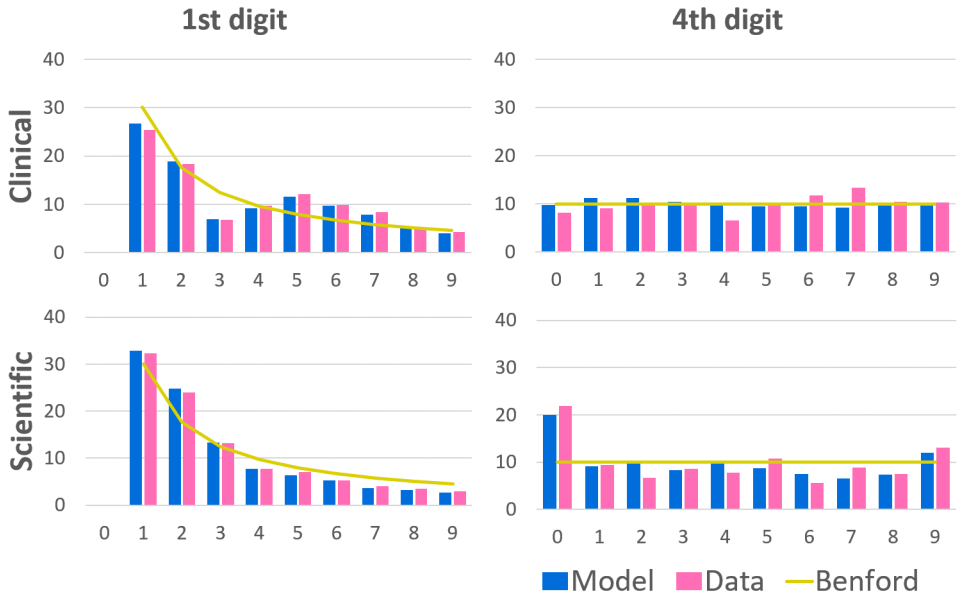

Figure 6.5: Distributions of significant digits from *d-RNN* model, from data, and from theoretical expectation (Benford's law).

are averaged over the development sets, from which all snippets also come. The *h-soft* strategy is selected mostly for integer numerals in the clinical (percentile points: '25', '50', '75'; typical drug dosage: '140') and scientific (years: '2004', '1997'; basis of scientific notation: '10') datasets. The *d-RNN* strategy is selected mostly for two-digit integers (dimensions in mm: '37') in the clinical dataset and for the NGC index for cataloguing astronomical objects (Dreyer, 1888) in the scientific dataset. The *MoG* strategy is selected mostly for non-integer numerals (end-diastolic volumes: '138.47') in the clinical dataset and for other indices for cataloguing astronomical objects, e.g. GL (Gliese, 1988) and HIP (Perryman et al., 1997). Most numerals for which the *MoG* strategy is selected are OOV.

6.1.5 Discussion

Findings Our most important findings are:

- Numerals are harder targets than other words for language models, as all models had a high adjusted perplexity for that subset (Table 6.2). This finding can be easily overlooked if one examines only the unadjusted perplexity (Ta-

Table 6.4: Examples of numerals for each strategy of the combination model. We show numerals most highly associated with each strategy (on average, in the development set) and text snippets that contain them.

	Clinical	Scientific
h-soft	**Numerals**: 50, 17, 100, 75, 25, 1, 140, 2012, 2010, 2011, 8, 5, 2009, 2013, 7, 6, 2, 3, 2008, 4... **Snippets**: *"late enhancement ($>$ **75** %)", "late gadolinium enhancement ($<$ **25** %)", "infarction (**2** out of **17** segments)", "infarct with **4** out of **17** segments nonviable", "adenosine stress perfusion @ **140** mcg", "stress perfusion (adenosine **140** mcg"*	**Numerals**: 1992, 2001, 1995, 2003, 2009, 1993, 2010, 1994, 1998, 2002, 2006, 1997, 2005, 1990, 10, 2008, 2007, 2004, 1983, 1991... **Snippets**: *"sharp et al . **2004**", "li et al . **2003**", "3.5 $\times$ **10**ˆ4", "0.3 $\times$ **10**ˆ16"*
d-RNN	**Numerals**: 42, 33, 31, 43, 44, 21, 38, 36, 46, 37, 32, 39, 26, 28, 23, 29, 45, 40, 49, 94... **Snippets**: *"aortic root is dilated (measured **37** x **37** mm", "ascending aorta is not dilated (**32** x **31** mm"*	**Numerals**: 294, 4000, 238, 6334, 2363, 1275, 2366, 602, 375, 1068, 211, 6.4, 8.7, 600, 96, 0.65, 700, 1.17, 4861, 270... **Snippets**: *"ngc **6334** stars", "ngc **2366** shows a wealth of small structures"*
MoG	**Numerals**: 74.5, 69.3, 95.9, 96.5, 72.5, 68.6, 82.1, 63.7, 78.6, 69.6, 69.5, 82.2, 68.3, 73.2, 63.2, 82.6, 77.7, 80.7, 70.7, 70.4... **Snippets**: *"stroke volume **46.1** ml", "stroke volume **65.6** ml", "stroke volume **74.5** ml", "end diastolic volume **82.6** ml", "end diastolic volume **99.09** ml", "end diastolic volume **138.47** ml"*	**Numerals**: 12961, 766, 7409, 4663, 44.3, 1819, 676, 1070, 5063, 323, 264, 163296, 2030, 77, 1.15, 196, 0.17, 148937, 0.43, 209458... **Snippets**: *"hip **12961** and gl **676** a are orbited by giant planets," "velocities of gl **676**", "velocities of hip **12961"***

ble 6.3), which does not account for the OOV-rate, or only perplexities evaluated on all tokens (Table 6.2), which is overwhelmed by the high proportion of non-numeral words (Table 6.1).

- Hierarchical modelling of numerals and words can drastically improve language modelling performance, as the hierarchical softmax models (*h-soft* and *hd-soft*) perform much better than their non-hierarchical variants (*soft* and *d-soft*, respectively; Table 6.2). This is possibly because the non-hierarchical softmax tries to separate the representations (output embeddings) of numerals

apart from those of words, which results in the representations of all numerals being too similar between one another and with the unknown numeral token (Figure 6.3); in a hierarchical approach, the numerals are already separated from other words, which allows for more degrees of freedom to discriminate between numerals. Moreover, hierarchical modelling allows for a different model to be used for each of two branches, i.e. to model numerals and to model words.

- Softmax models with digit-based embeddings for numerals (*d-soft* and *hd-soft*) do not perform much better (if any better at all) than softmax models with only tokens-based embeddings (*soft* and *h-soft*, respectively; Table 6.2). This is possibly because language modelling performance is bounded by low performance on the OOV tokens, and digit-based embeddings fail to address that issue (the 'UNK' and 'UNKNUM' tokens are still required).

- Compositional modelling (based on digit-based composition) in the hierarchical branch of numerals can further improve language modelling performance, as the *d-RNN* model performed better than the softmax models (*h-soft* and *hd-soft*) in both datasets (Table 6.2). This is possibly because *d-RNN* supports an open vocabulary of numerals, which removes the need for an 'UNKNUM' token in the vocabulary.

- Continuous density modelling (based on mixture of Gaussians) in the hierarchical branch of numerals may or may not improve language modelling performance, as *MoG* performed better than the softmax models (*h-soft* and *hd-soft*) in the clinical dataset, but worse in the scientific dataset (Table 6.2). This is possibly because, although *MoG* supports an open vocabulary of numerals, its performance depends on a good selection for the parameters of its mixture components (means and variances).

- A combination of these strategies (categorical softmax, composition and continuous densities) might be the most appropriate for the modelling of numerals, as the *combination* model that implements all strategies (*h-soft*, *d-RNN*,

and *MoG*, respectively) had the best overall performance (Table 6.2). This is possibly because a different strategy needs to be selected for each numeral, which can be inferred from its context (Table 6.4).

Limitations There are several limitations for this experiment:

- **Limitations from Data** Because of out tokenisation, all numerals represent non-negative numbers; this means that the token-level RNN should learn when a number should be negative. Numerals appearing in the scientific texts can be very large, e.g. it contains an integer with 26 digits; this leads to numerical instabilities, and it can also dominate numerical averages, as in the cases of the mean of the date and of the absolute errors. Furthermore, the scientific text contains numerals from several numeral systems with overlapping forms, e.g. '10' is at the same type the decimal (base-10) for 10, the binary (base-2) for 2, the octal (base-8) for 8, etc.

- **Limitations from Models** Each of the models has its own limitations: softmax models (*soft*, *d-soft*, *d-soft*, and *hd-soft*) try to enforce a categorical distribution on numerals that might represent continuous quantities; the *d-RNN* tries to learn the numeral system, but the data can contain several numeral systems; the *MoG* has a performance that largely depends on the means and variances of its components; and the *combination* needs to learn the optimal strategy in an unsupervised way and might make paradoxical selections at test time.

6.2 Predicting Numbers with Language Models

6.2.1 Approach

The evaluation with (adjusted) perplexities in Section 6.1. is concerned with symbolic performance on numerals; in this section, we investigate the task of next number prediction and evaluate on the magnitude of numbers, which is their most prominent semantic content (Dehaene et al., 2003; Dehaene and Cohen, 1995).

6.2.2 Data

We use the clinical dataset used throughout the thesis and the scientific dataset introduced in Section 6.1

6.2.3 Experimental Setup

Models We consider the following models for making predictions for the next number

1. **mean** This model predicts with the *mean* of the training data;
2. **median** This model predicts with the *median* of the training data.
3. **soft, d-soft, h-soft, hd-soft, d-RNN, MoG, and combination** This model uses a LM (defined in Section 6.1) to score a set of candidate numerals and predicts with the number of the best scoring numeral. To create the set of candidate numerals, we first create a set of candidate numbers that is the union of in-vocabulary numbers and 100 percentile points from the training set. We generate the set of candidate numerals from these numbers by converting them into numerals by considering all formats up to n decimal points. We select n to represent 90% of numerals seen in the training data, which yields $n = 3$ and $n = 4$ for the clinical and scientific data, respectively.

Training Details All language models are trained as in Section 6.1.

6.2.4 Results

Evaluation on the Number Line Table 6.5 shows regression evaluation with absolute (RMSE, MAE, and MdAE) and relative (MAPE) error metrics for the various models and for two naive baselines (*mean* and *median*). All results are evaluated on the test set. The RMSE and MAE of the scientific dataset are in the order of 10^9. Absolute errors (RMSE, MAE, and MdAE) are not be comparable between datasets, because they are scale-dependent; in fact, unless the target numbers are assumed to be dimensionless quantities (i.e. pure numbers without units), absolute errors might be ill-defined, because they are unit-dependent. Such difficulties with comparability or definition do not occur for the MAPE metric, which is a relative

Table 6.5: Results for regression evaluated on the test set for the clinical and scientific data. Best performances in **bold**; performances equal to or worse than the *mean* or *median* baseline in *italics*.

	Clinical				Scientific	
model	**RMSE**	**MAE**	**MdAE**	**MAPE%**	**MdAE**	**MAPE%**
mean	*1043.68*	*294.95*	*245.59*	*2353.11*	$\sim 10^{20}$	$\sim 10^{23}$
median	*1036.18*	*120.24*	*34.52*	*425.81*	*4.20*	*8039.15*
soft	997.84	80.29	12.70	*621.78*	3.00	1947.44
d-soft	991.38	74.44	13.00	*503.57*	3.50	*15208.37*
h-soft	*1095.01*	*167.19*	14.00	*746.50*	3.00	1652.21
hd-soft	1001.04	83.19	12.30	*491.85*	3.00	2703.49
d-RNN	1009.34	70.21	9.00	*513.81*	3.00	1287.27
MoG	998.78	**57.11**	**6.92**	**348.10**	**2.10**	**590.42**
combination	**989.84**	69.47	9.00	*552.06*	3.00	2332.50

error that is independent of scale and units. According to MAPE, the *MoG* model achieves the best performance in both datasets, and it is the only model to perform better than the *median* baseline for the clinical data. Finally, the *combination* model does not achieve a better MAPE than any of its individual components (*MoG*, *d-RNN*, *hd-soft*).

Probability of the Next Numeral Figure 6.6 visualises the probabilities of the various models for the next numeral in two examples from the development set of the clinical data. The lines connect the the probabilities of numerals with the same number of decimal digits. For the *h-soft* model, the probabilities are spiked; for *d-RNN*, they follow a saw-tooth pattern; and, for *MoG*, they are smooth with an occasional spike, whenever a narrow component allows for it.

6.2.5 Discussion

Findings Our most important findings are:

- Continuous density modelling (based on mixture of Gaussians) of numerals can drastically improve the number prediction performance of language models, as the language model that implemented that strategy (*MoG*) performed better than models that implemented other strategies or a combination of them (Table 6.5). This is possibly because the smooth output probabilities of the

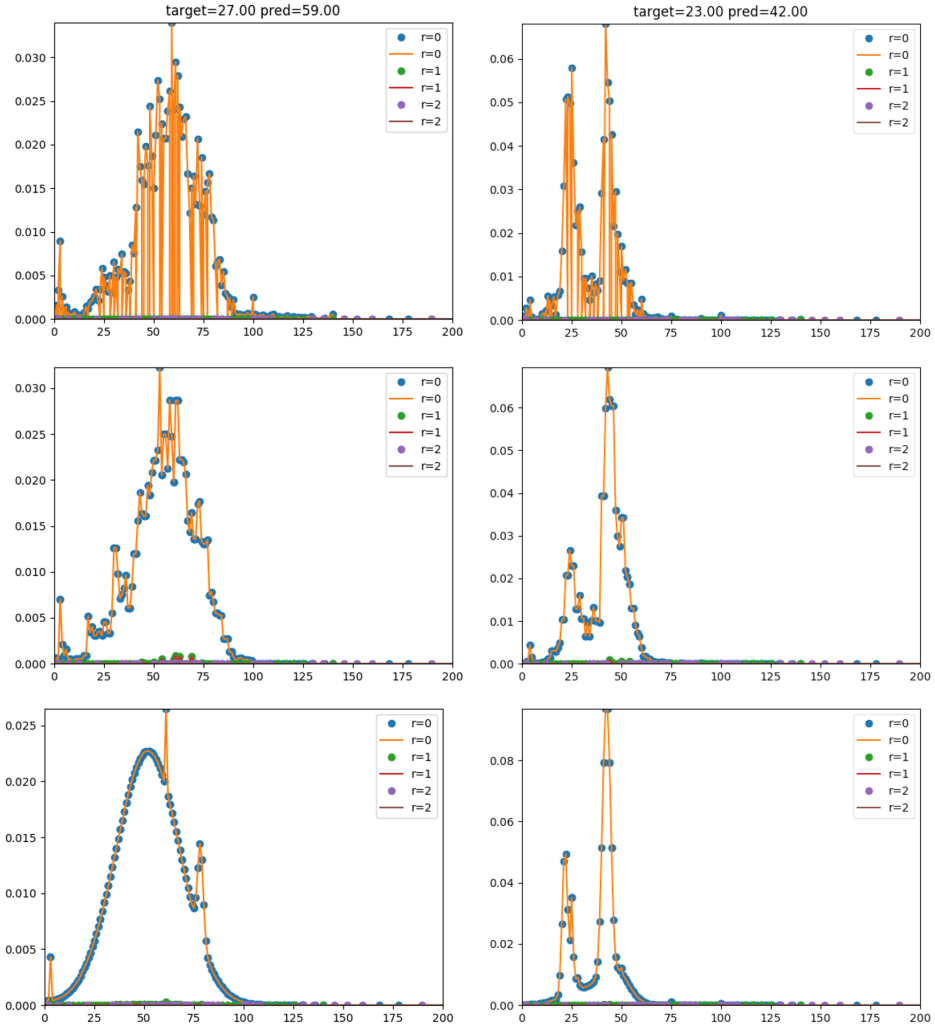

Figure 6.6: Example model predictions for the *h-soft* (top), *d-RNN* (middle) and *MoG* (bottom) models. Examples from the clinical development set.

MoG model (Figure 6.6) allow it to generalise better.

Limitations Additionally to the limitations from Section 6.2, there are additional limitations for this experiment:

- **Limitations from Evaluation** Absolute errors (RMSE, MAE, and MdAE) are not be comparable between datasets, because they are scale-dependent; in fact, unless the target numbers are assumed to be dimensionless quantities (i.e. pure numbers without units), absolute errors might be ill-defined, because they are unit-dependent. While most of these issues are solved by

the use of relative error metrics (MAPE), there still exist numerals for which evaluation with numerical errors is not meaningful: numerals that are simply labels and their magnitude is irrelevant, e.g. phone numbers.

Chapter 7

Conclusion

'That's Numberwang!'
— Robert Webb as 'Mr Terrific',
in *That Mitchell and Webb Look*

Recapitulation This thesis was about the joint modelling of data of different types: words, numbers, and numerals. We motivated and formulated the research questions (Chapter 1), reviewed the background of tasks and models (Chapter 2), and positioned this work within the broader literature by discussing related work (Chapter 3). We addressed the research questions by setting up and investigating experiments for a variety of models: word classification with numbers as inputs and number regression with words as inputs (Chapter 4); language models with numbers as inputs and their downstream applications in text prediction and semantic error detection and correction (Chapter 5); and language models to model numerals and to predict numbers (Chapter 6). In this chapter, we conclude with an overview of the contributions of the thesis, critical reflection, and suggestions for future research.

Major Methodological Contributions The most important methodological innovations of this work are as follows:

- We contribute to modelling by proposing the use of numbers as inputs to language models and the mechanisms to achieve that (magnitude-dependent embeddings and conditioning on lexicalised KB). We find that this can improve performance for language modelling and several downstream applications (text prediction and semantic error detection and correction) (Chapter 5).

- We contribute to modelling by proposing a variety of strategies (categorical softmax, hierarchical, digit-by-digit compositional, from continuous probability density, and combination of strategies) that improve the ability of language models to predict numbers and to model numerals (Chapter 6).

- We contribute to evaluation by using perplexities that are adjusted and evaluated separately for the subset of numerals, to account for varying OOV-rates and for the dominating prevalence of words in the data (Chapters 5 and 6). We also use evaluations on the number line for numerical outputs of language models (Chapters 6). These evaluations show a more faithful picture of the behaviour of language models on the output of numerals and numbers, and can be used to measure improvements.

Major Findings The most important findings of this work are the following:

- Combining words and numbers as inputs to word classification and number regression task can improve over using inputs of a single type (Chapter 4).

- Numerals are harder targets than other words for language models. This finding is not immediately evident, but requires to evaluate with adjusted perplexity only on the subset of numerals (Chapters 5 and 6).

- Numbers as inputs to language models (using magnitude-dependent embeddings and conditioning on a lexicalised KB) can improve the performance for language modelling and for several downstream applications (text prediction and semantic error detection and correction) (Chapter 5).

- Modelling numerals and predicting numbers with language models might benefit from using different strategies (categorical softmax, hierarchical, digit-by-digit compositional, from continuous probability density, and combination of strategies) for different contexts. For modelling numerals, a combination of strategies was found to have the best performance for (adjusted) perplexity; for predicting numbers, the continuous probability density strat-

egy with a mixture of Gaussians achieved the best relative numerical error (Chapter 6).

7.1 Critical Reflection

The findings and contributions should be considered along the following points of critical reflection:

- **Domain Dependence** All our analyses, experiments, and investigations were executed on texts from the clinical and scientific domains, which are expert technical domains with their particular idiolects and idiosyncracies. Although the results mostly agreed between the two domains, whenever compared, generalisation of the proposed techniques is not guaranteed for different domains, e.g. non-technical texts.

- **Simulation versus Real-Life Application** Our evaluations for text prediction (word prediction and word completion) and semantic error detection and correction were simulated under the assumption of a perfect user, one who is engaged, cooperative, consistent, and immediately responsive to the systems. In a real-life application, the true performance of such a system would also depend on the variation among users, the system design, and other external factors. Furthermore, our evaluation for text prediction relied on the correctness of the texts in the data, which cannot be guaranteed, and for error correction on the distribution of the simulated errors.

- **Numerical Instabilities** This problem refers to numerical overflows and indefinite operations that might arise when handling numerical data with computers, e.g. multiplying two large numbers, dividing by zero, etc. For most applications, numbers refer to a predetermined set of attributes and are often normalised for that attribute's numerical range, so as to avoid numerical instabilities; however, we had no control over the set of possible attributes or range of numbers in text data, e.g. we encountered an integer in the scientific data that was 26 digits long. The problem of numerical instabilities was a concern

throughout this work and had a major influence on the design decisions for the models, e.g. we consciously tried to avoid non-bounded operations that involved numbers from the text.

- **Predictability of Numbers** The distribution for the next numeral in a text might depend on information that is not yet available, e.g. the distribution depends on the units for the number, which typically come after the numeral. These make the distribution for the next number inherently multimodal, and any evaluation against point estimates might be too harsh. A solution would be to predict from the probability of the whole document for different substitutions of the numeral candidates; however, this was beyond the scope of this thesis. A further complication relates to the alternative function of numbers as labels, e.g. the magnitude of a phone number is irrelevant. It might be desirable to exclude such numerals from the evaluation of number prediction.

- **Challenges of Application** Translating the findings from any simulation into a practical clinical application is subject to challenges: the implementation of the model introduces software engineering challenges (userability of human-computer interface, scalability to large datasets, security of sensitive information, etc.), and the human end-user might exhibit different behaviours (because of preferences, habits, idiosyncrasy, etc.) from the simulated ideal user (Chapman et al., 2011). Mismatched expectations, poor implementation, or inadequate integration might inhibit the adoption of the clinical application. Additional challenges are posed by the variability of sublanguages, intended user groups, and support goals (Demner-Fushman et al., 2009) that can differ between and within clinical institutions.

- **Opportunities of Application** Technological advances and clinical needs create opportunities for translating this work into clinical applications: electronic health records are becoming increasingly more widespread in health care (Blumenthal and Tavenner, 2010); clinicians need to enter the patient's data quickly and accurately, and they need to be able to switch between

structured and unstructured documentation, depending on their needs (Rosenbloom et al., 2011); advances in computing (cloud computing, GPUs, etc) allow us to develop, train, and serve complex models (e.g. neural networks) to user-interfaces in computers, tablets, smart phones, etc.; finally, large-scale, publicly available clinical datasets, e.g. MIMIC-III (Johnson et al., 2016), are being released and can be used for training such models.

7.2 Future Work

We conclude this thesis with suggestions for future research, based on our insights and on the trends in the literature. Our suggestions are as follows:

- Validate and extend the findings by experimenting in other sub-domains of scientific and clinical data (e.g. different specialty or different hospital) or other number-intensive domains (e.g. financial documents).

- Translate the techniques from this work and evaluate as a practical application in an actual setting, e.g. deploy a text editor with predictive text entry and error correction functionalities in a hospital and assess with operational evaluation metrics, such as time and money savings, reduction of clinically significant errors, user attitudes towards the application, etc.

- Validate and extend the findings by experimenting with other number-intensive applications of language models, e.g. optical character recognition (OCR), voice recognition, text generation, etc.

- Adapt and apply techniques from this work for models in other number-related tasks, e.g. numerical question answering, numerical information extraction, numerical entailment, commonsense reasoning, etc.

- Extend the models by adding new or refining existing strategies for numerical output, e.g. from continuous probability density functions with different distributions.

- Extend the abilities of the models by injecting mathematical background knowledge for numerical reasoning, linear solvers, algebraic reasoning, numerical expressions between attributes, etc.

Appendix A

Clinical Dataset

The dataset comprises 16,015 anonymised clinical records from the London Chest Hospital. Each record has a knowledge base (KB) of attribute-value pairs and a text report about the patient.

Statistical Description of Patients Table A.1 shows all the attributes from the KB in the clinical dataset and their statistical description (mean, standard deviation, and percentage of the records where the value of the attribute is reported). The average patient is 56.46 years old, and 66% of the patients are male. The other attributes refer to measurements of the end-diastolic volume (EDV), end-systolic volume (ESV), ejection fraction (EF), stroke volume (SV), heart rate (HR), cardiac output (CO), and mass in relation to the left ventricle (LV) or the right ventricle (RV). Repeated measurements of an attribute are denoted with numbers, e.g. LV2 EDV refers to the second (follow-up) measurement of the end-diastolic volume of the left ventricle.

Statistical Description of Texts The average length of the clinical text documents is about 207 tokens, 4.3% of which are numerals. A more detailed description of the statistics of the text documents can be found in Tables 5.3 and 6.1.

Workflow The data for each patient are collected following a workflow as below:

1. A clinician requests that the cardiac structure and function of the patient be assessed. This decision is based on clinical indications, e.g. their history, results of diagnostic tests, clinical findings, etc.

Table A.1: Statistics for patients in clinical dataset. Mean, standard deviation, and percentage of records that have a value for that attribute.

	mean	std	found in % of records
sex='male'	66%	-	100
age	56.46	15.53	100
LV1 EDV	164.62	62.03	63
LV1 ESV	79.11	57.89	63
LV1 EF	55.85	15.21	63
LV1 SV	85.58	25.39	62
LV1 HR	71.17	14.17	18
LV1 Mass	132.58	51.27	4
LV1 CO	5.84	1.67	3
LV2 EDV	182.88	70.50	5
LV2 ESV	94.22	60.16	5
LV2 EF	52.00	13.70	5
LV2 SV	90.13	26.28	5
LV2 Mass	131.78	41.88	1
LV2 CO	4.96	1.40	1
RV1 EDV	176.60	96.62	1
RV1 ESV	82.54	46.35	1
RV1 EF	54.59	11.69	1
RV1 SV	91.72	32.65	1
RV1 CO	6.49	0.97	1

2. The patient undergoes cardiovascular magnetic resonance (CMR),[1] which is the gold standard method for the assessment of cardiac structure and function.

3. The CMR scans are analysed using the cvi[42] post-processing tool (Circle Cardiovascular imaging, Calgary, Canada). The process of the analysis involves visually identifying the systolic and diastolic phase (see Table 4.1 for an overview of the cardiac cycle) and annotating contours in the images (annotation is assisted by the post-processing tool with the option of manual correction). The post-processing combines the annotations provided by the analyser with other automatically extracted features to populate the relevant attributes in the patient's KB, e.g. EDV, ESV, EF, mass, etc.

4. The analysis of the CMR is finalised, and the clinician composes the text

[1] CMR is also known as cardiac magnetic resonance imaging (cardiac MRI)

Table A.2: Number of CMRs requested, analysed, or finalised per person. The names have been anonymised. Note that some analyses might have been requested, analysed, or finalised by more than one person.

Requested by	#CMRs	Analysed by	#CMRs	Finalised by	#CMRs
not available	829	anonymised_A	2504	anonymised_A	6779
anonymised_1	511	anonymised_B	1473	anonymised_B	3275
anonymised_2	407	anonymised_4	1068	not available	2146
anonymised_3	379	anonymised_D	1033	anonymised_C	2110
anonymised_D	373	anonymised_E	972	anonymised_D	1315
anonymised_A	319	anonymised_C	854	anonymised_E	598
other	13403	other	8982	other	241

of the report. The clinician has access to the values from the KB and the previous clinical notes on the patient (e.g. history), and can directly view the CMR scans, if they wish. Often, clinicians use templates, copy text from the patient's record, or copy text from records of other patients (Weir et al., 2003).

5. At various points in the workflow, the process might be reviewed by multiple people, e.g. in the case of significant discrepancies. As a result, for our dataset each of the CMRs was requested on average by 1.01 (standard deviation 0.12), analysed by 1.05 (std 0.24), and finalised by 1.03 (std 0.17) persons. A more detailed (yet anonymised) view of the number of CMRs processed per person can be found in Table A.2.

Bibliography

S. Afantenos, V. Karkaletsis, and P. Stamatopoulos. Summarization from medical documents: a survey. *Artificial intelligence in medicine*, 33(2):157–177, 2005.

S. Ahn, H. Choi, T. Pärnamaa, and Y. Bengio. A neural knowledge language model. *arXiv preprint arXiv:1608.00318*, 2016.

E. Aramaki, T. Imai, K. Miyo, and K. Ohe. Uth: Svm-based semantic relation classification using physical sizes. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 464–467. Association for Computational Linguistics, 2007.

A. R. Aronson. Effective mapping of biomedical text to the umls metathesaurus: the metamap program. In *Proceedings of the AMIA Symposium*, page 17. American Medical Informatics Association, 2001.

D. G. Arts, N. F. De Keizer, and G.-J. Scheffer. Defining and improving data quality in medical registries: a literature review, case study, and generic framework. *Journal of the American Medical Informatics Association*, 9(6):600–611, 2002.

R. Bartsch. The grammar of relative adjectives and comparison. In *Formal Aspects of Cognitive Processes*, pages 168–185. Springer, 1975.

H. Bast and I. Weber. Type less, find more: fast autocompletion search with a succinct index. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, 2006.

A. Belz. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, 14(4):431–455, 2008.

F. Benford. The law of anomalous numbers. *Proceedings of the American philosophical society*, pages 551–572, 1938.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

A. Berger and J. Lafferty. Information retrieval as statistical translation. In *ACM SIGIR Forum*, volume 51, pages 219–226. ACM, 2017.

D. Beukelman and P. Mirenda. *Augmentative and alternative communication*. Brookes, 2005.

S. Bickel, P. Haider, and T. Scheffer. Predicting sentences using n-gram language models. In *Proceedings of Human Language Technology and Empirical Methods in Natural Language Processing*, 2005.

Z. Bitvai and T. Cohn. Non-linear text regression with a deep convolutional neural network. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 180–185, 2015.

D. Blumenthal and M. Tavenner. The "meaningful use" regulation for electronic health records. *New England Journal of Medicine*, 363(6):501–504, 2010.

T. Boroş, S. D. Dumitrescu, A. Zafiu, V. B. Mititelu, and I. P. Vaduva. Racai gec–a hybrid approach to grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 43–48, 2014.

S. Bowman. Impact of Electronic Health Record Systems on Information Integrity: Quality and Safety Implications. *Perspectives in Health Information Management*, page 1, 2013.

R. Brandenburg, E. Chazov, G. Cherian, A. Falase, Y. Grosgogeat, C. Kawai, F. Loogen, V. M. Judez, E. Orinius, J. Goodwin, et al. Report of the who/isfc task force on definition and classification of cardiomyopathies. *Circulation*, 64 (2):437A–438A, 1981.

C. Brown. Assistive technology computers and persons with disabilities. *Communications of the ACM*, 35(5):36–45, 1992.

P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311, 1993.

E. Bruni, N.-K. Tran, and M. Baroni. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47, 2014.

J. I. Campbell. *Handbook of mathematical cognition*. Psychology Press, 2005.

M. Cannataro, O. Alfieri, and F. Fera. Knowledge-based compilation of magnetic resonance diagnosis reports in neuroradiology. In *25th International Symposium on Computer-Based Medical Systems (CBMS)*. IEEE, 2012.

A. Carlberger, J. Carlberger, T. Magnuson, M. S. Hunnicutt, S. E. Palazuelos-Cagigas, and S. A. Navarro. Profet, a new generation of word prediction: An evaluation study. In *Proceedings, ACL Workshop on Natural language processing for communication aids*, pages 23–28, 1997.

A. T. Chaganty and P. Liang. How much is 131 million dollars? putting numbers in perspective with compositional descriptions. *arXiv preprint arXiv:1609.00070*, 2016.

V. Chahuneau, K. Gimpel, B. R. Routledge, L. Scherlis, and N. A. Smith. Word salad: Relating food prices and descriptions. In *Proceedings of the 2012 Joint*

Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, pages 1357–1367. Association for Computational Linguistics, 2012.

Y.-C. Chang, J. S. Chang, H.-J. Chen, and H.-C. Liou. An automatic collocation writing assistant for taiwanese efl learners: A case of corpus-based nlp technology. *Computer Assisted Language Learning*, 21(3):283–299, 2008.

W. W. Chapman, P. M. Nadkarni, L. Hirschman, L. W. D'Avolio, G. K. Savova, and O. Uzuner. Overcoming barriers to nlp for clinical text: the role of shared tasks and the need for additional creative solutions. *J Am Med Inform Assoc*, 18(5), 2011.

E. Charniak. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 124–131. Association for Computational Linguistics, 2001.

C. Chelba and F. Jelinek. Exploiting syntactic structure for language modeling. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 225–231. Association for Computational Linguistics, 1998.

C. Chelba and F. Jelinek. Structured language modeling. *Computer Speech & Language*, 14(4):283–332, 2000.

C.-H. Chen, S.-H. Hsieh, Y.-S. Su, K.-P. Hsu, H.-H. Lee, and F. Lai. Design and implementation of web-based discharge summary note based on service-oriented architecture. *Journal of medical systems*, 36(1):335–345, 2012.

D. L. Chen and R. J. Mooney. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135. ACM, 2008.

S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–394, 1999.

W. Chen, D. Grangier, and M. Auli. Strategies for training large vocabulary neural language models. *arXiv preprint arXiv:1512.04906*, 2015.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of EMNLP*, 2014.

M. Chodorow, M. Gamon, and J. Tetreault. The utility of article and preposition error correction systems for English language learners: Feedback and assessment. *Language Testing*, 2010.

J. Chu-Carroll, D. A. Ferrucci, J. M. Prager, and C. A. Welty. Hybridization in question answering systems. *New Directions in Question Answering*, 3:116–121, 2003.

K. W. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *International Conference on Acoustics, Speech, and Signal Processing,*, pages 695–698. IEEE, 1989.

I. Dagan, D. Roth, M. Sammons, and F. M. Zanzotto. Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220, 2013.

D. Dahlmeier and H. T. Ng. Correcting semantic collocation errors with l1-induced paraphrases. In *Proceedings of EMNLP*, pages 107–117, 2011.

R. Dale and A. Kilgarriff. Helping our own: The hoo 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 242–249. Association for Computational Linguistics, 2011.

R. Dale, I. Anisimoff, and G. Narroway. Hoo 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 54–62. Association for Computational Linguistics, 2012.

D. Davidov and A. Rappoport. Extraction and approximation of numerical attributes from the web. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1308–1317. Association for Computational Linguistics, 2010.

R. De Felice and S. G. Pulman. A classifier-based approach to preposition and determiner error correction in l2 english. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 169–176. Association for Computational Linguistics, 2008.

M.-C. de Marneffe, C. D. Manning, and C. Potts. Was it good? it was provocative. learning the meaning of scalar adjectives. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 167–176. Association for Computational Linguistics, 2010.

S. Dehaene and L. Cohen. Towards an anatomical and functional model of number processing. *Mathematical cognition*, 1(1):83–120, 1995.

S. Dehaene, M. Piazza, P. Pinel, and L. Cohen. Three parietal circuits for number processing. *Cognitive neuropsychology*, 20(3-6):487–506, 2003.

D. Demner-Fushman, W. W. Chapman, and C. J. McDonald. What can natural language processing do for clinical decision support? *Journal of biomedical informatics*, 42(5):760–772, 2009.

J. C. Denny, A. Spickard III, P. J. Speltz, R. Porier, D. E. Rosenstiel, and J. S. Powers. Using natural language processing to provide personalized learning opportunities from trainee clinical notes. *Journal of biomedical informatics*, 56: 292–299, 2015.

J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

M. S. Donaldson, J. M. Corrigan, L. T. Kohn, et al. *To err is human: building a safer health system*, volume 6. National Academies Press, 2000.

J. L. E. Dreyer. A new general catalogue of nebulæ and clusters of stars, being the catalogue of the late sir john fw herschel, bart, revised, corrected, and enlarged. *Memoirs of the Royal Astronomical Society*, 49:1, 1888.

M. D. Dunlop and A. Crossan. Predictive text entry methods for mobile phones. *Personal Technologies*, 4(2-3):134–143, 2000.

A. Emami and F. Jelinek. A neural syntactic language model. *Machine learning*, 60(1-3):195–227, 2005.

J. Eng and J. M. Eisner. Informatics in radiology (inforad) radiology report entry with automatic phrase completion driven by language modeling. *Radiographics*, 24(5):1493–1501, 2004.

D. A. Evans, N. D. Brownlow, W. R. Hersh, and E. M. Campbell. Automating concept identification in the electronic medical record: an experiment in extracting dosage information. In *Proceedings of the AMIA Annual Fall Symposium*, page 388. American Medical Informatics Association, 1996.

A. Fazly and G. Hirst. Testing the efficacy of part-of-speech information in word completion. In *Proceedings of the EACL 2003 Workshop on Language Modeling for Text Entry Methods*, 2003.

M. Felice and Z. Yuan. Generating artificial errors for grammatical error correction. In *Proceedings of EACL*, pages 116–126, 2014.

M. Felice, Z. Yuan, Ø. E. Andersen, H. Yannakoudakis, and E. Kochmar. Grammatical error correction using hybrid systems and type filtering. In *CoNLL Shared Task*, pages 15–24, 2014.

K. Filippova, E. Alfonseca, C. A. Colmenares, L. Kaiser, and O. Vinyals. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on*

Empirical Methods in Natural Language Processing, EMNLP 2015, pages 360–368, 2015.

J. R. Firth. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*, 1957.

M. Fleischman and D. Roy. Grounded Language Modeling for Automatic Speech Recognition of Sports Video. In *Proceedings of ACL*, pages 121–129, 2008.

M. Fontoura, R. Lempel, R. Qi, and J. Zien. Inverted index support for numeric search. *Internet Mathematics*, 3(2):153–185, 2006.

G. Foster, P. Langlais, and G. Lapalme. User-friendly text prediction for translators. In *Proceedings of the ACL-02 conference on Empirical methods in natural language*, 2002.

L. Frazier, C. Clifton Jr, and B. Stolterfoht. Scale structure: Processing minimum standard and maximum standard scalar adjectives. *Cognition*, 106(1):299–324, 2008.

C. Friedman, P. O. Alderson, J. H. Austin, J. J. Cimino, and S. B. Johnson. A general natural-language text processor for clinical radiology. *Journal of the American Medical Informatics Association*, 1(2):161–174, 1994.

Y. Futagi, P. Deane, M. Chodorow, and J. Tetreault. A computational approach to detecting collocation errors in the writing of non-native speakers of english. *Computer Assisted Language Learning*, 21(4):353–367, 2008.

M. Gambhir and V. Gupta. Recent automatic text summarization techniques: a survey. *Artif. Intell. Rev.*, 47(1):1–66, 2017.

M. Gamon. Using Mostly Native Data to Correct Errors in Learners' Writing: A Meta-Classifier Approach. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 163–171. Association for Computational Linguistics, 2010.

M. Gamon, J. Gao, C. Brockett, A. Klementiev, W. B. Dolan, D. Belenko, and L. Vanderwende. Using Contextual Speller Techniques and Language Modeling for ESL Error Correction. In *IJCNLP*, volume 8, pages 449–456, 2008.

S. Gao, X. Yang, Z. Yu, X. Pan, and J. Guo. Chinese-naxi machine translation method based on naxi dependency language model. *Int. J. Machine Learning & Cybernetics*, 8(1):333–342, 2017.

N. Garay-Vitoria and J. Abascal. Text prediction systems: a survey. *Universal Access in the Information Society*, 4(3):188–203, 2006.

N. Garera and D. Yarowsky. Modeling latent biographic attributes in conversational genres. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 710–718. Association for Computational Linguistics, 2009.

F. A. Gers and E. Schmidhuber. Lstm recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks*, 12(6): 1333–1340, 2001.

F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12:2451–2471, 1999.

F. A. Gers, N. N. Schraudolph, and J. Schmidhuber. Learning precise timing with lstm recurrent networks. *Journal of machine learning research*, 3(Aug):115–143, 2002.

S. Ghosh, O. Vinyals, B. Strope, S. Roy, T. Dean, and L. Heck. Contextual lstm (clstm) models for large scale nlp tasks. *arXiv:1602.06291*, 2016.

W. Gliese. The third catalogue of nearby stars. *Stand. Star Newsl. 13, 13*, 13, 1988.

S. Goldberg, A. Niemierko, and A. Turchin. Analysis of data errors in clinical research databases. In *AMIA*. Citeseer, 2008.

Y. Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

Y. Gong, L. Hua, and S. Wang. Leveraging user's performance in reporting patient safety events by utilizing text prediction in narrative data entry. *Computer methods and programs in biomedicine*, 131:181–189, 2016.

A. Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

A. Graves and N. Jaitly. Towards end-to-end speech recognition with recurrent neural networks. In *International Conference on Machine Learning*, pages 1764–1772, 2014.

N. Green. Representing normative arguments in genetic counseling. In *AAAI Spring Symposium: Argumentation for Consumers of Healthcare*, pages 64–68, 2006.

J. Gu, Z. Lu, H. Li, and V. O. Li. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*, 2016.

C. Gulcehre, S. Ahn, R. Nallapati, B. Zhou, and Y. Bengio. Pointing the unknown words. *arXiv preprint arXiv:1603.08148*, 2016.

Ç. Gülçehre, O. Firat, K. Xu, K. Cho, and Y. Bengio. On integrating a language model into neural machine translation. *Computer Speech & Language*, 45:137–148, 2017.

S. Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42 (1-3):335–346, 1990.

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

S. D. Hdez and H. Calvo. Conll 2014 shared task: Grammatical error correction with a syntactic n-gram language model from a big corpora. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 53–59, 2014.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

R. Hoffmann, C. Zhang, and D. S. Weld. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 286–295. Association for Computational Linguistics, 2010.

P. Holmes-Higgin. Text generation—using discourse strategies and focus constraints to generate natural language text by kathleen r. mckeown, cambridge university press, 1992, pp 246,£ 13.95, isbn 0-521-43802-0. *The Knowledge Engineering Review*, 9(4):421–422, 1994.

M. J. Hosseini, H. Hajishirzi, O. Etzioni, and N. Kushman. Learning to solve arithmetic word problems with verb categorization. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 523–533, 2014.

E. Hovy, U. Hermjakob, C.-Y. Lin, and D. Ravichandran. Using knowledge to facilitate factoid answer pinpointing. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics, 2002.

L. Hua, S. Wang, and Y. Gong. Text prediction on structured data entry in healthcare: A two-group randomized usability study measuring the prediction impact on user performance. *Applied Clinical Informatics*, 5:249–263, 2014.

Y. Huang and H. J. Lowe. A novel hybrid approach to automated negation detection in clinical radiology reports. *Journal of the American Medical Informatics Association*, 14(3):304–311, 2007.

A. Iftene and M.-A. Moruz. Uaic participation at rte-6. 2010.

Institute for Safe Medication Practices. Ismp's list of error-prone abbreviations, symbols, and dose designations, 2017. URL `https://www.ismp.org/recommendations/error-prone-abbreviations-list`.

A. Intxaurrondo, E. Agirre, O. L. De Lacalle, and M. Surdeanu. Diamonds in the rough: Event extraction from imperfect microblog data. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 641–650, 2015.

P. Jacquemart and P. Zweigenbaum. Towards a medical question-answering system: a feasibility study. *Studies in health technology and informatics*, 2003.

S. Jean, K. Cho, R. Memisevic, and Y. Bengio. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*, 2014.

F. Jelinek. *Statistical methods for speech recognition*. MIT press, 1997.

M. Jeong, B. Kim, and G. G. Lee. Semantic-oriented error correction for spoken query processing. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, pages 156–161. IEEE, 2003.

M. Jeong, B. Kim, and G. G. Lee. Using higher-level linguistic knowledge for speech recognition error correction in a spoken q/a dialog. In *Proceedings of the HLT-NAACL 2004 Workshop on Spoken Language Understanding for Conversational Systems and Higher Level Linguistic Information for Speech Processing*, 2004.

A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

S. B. Johnson, S. Bakken, D. Dine, S. Hyun, E. Mendonça, F. Morrison, T. Bright, T. Van Vleck, J. Wrenn, and P. Stetson. An electronic health record based on structured narrative. *Journal of the American Medical Informatics Association*, 15(1):54–64, 2008.

M. Joshi, D. Das, K. Gimpel, and N. A. Smith. Movie reviews and revenues: An experiment in text regression. In *Human Language Technologies: The 2010 Annual*

Conference of the North American Chapter of the Association for Computational Linguistics, pages 293–296. Association for Computational Linguistics, 2010.

R. Józefowicz, W. Zaremba, and I. Sutskever. An empirical exploration of recurrent network architectures. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, pages 2342–2350, 2015.

R. C. Kadosh and A. Dowker. *The Oxford handbook of numerical cognition*. Oxford Library of Psychology, 2015.

D. Kasper, E. Braunwald, S. H. Fauci, D. Longo, and J. Jameson. Harrison's principles of internal medicine, 2005.

C. Kennedy. *Projecting the adjective: The syntax and semantics of gradability and comparison*. Routledge, 1999.

C. Kennedy. Vagueness and grammar: The semantics of relative and absolute gradable adjectives. *Linguistics and philosophy*, 30(1):1–45, 2007.

D. Kiela and S. Clark. Multi- and Cross-Modal Semantics Beyond Vision: Grounding in Auditory Perception. In *Proceedings of EMNLP*, pages 2461–2470, 2015. URL `http://aclweb.org/anthology/D15-1293`.

D. Kiela, L. Bulat, and S. Clark. Grounding Semantics in Olfactory Perception. In *Proceedings of ACL*, pages 231–236, 2015. URL `http://www.aclweb.org/anthology/P15-2038`.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

R. Kneser and H. Ney. Improved clustering techniques for class-based statistical language modelling. In *Third European Conference on Speech Communication and Technology*, 1993.

K. Knight and D. Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107, 2002.

E. Kochmar and E. J. Briscoe. Capturing anomalies in the choice of content words in compositional distributional semantic space. 2013.

S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. A. Smith. Predicting risk from financial reports with regression. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 272–280. Association for Computational Linguistics, 2009.

R. Koncel-Kedziorski, H. Hajishirzi, A. Sabharwal, O. Etzioni, and S. D. Ang. Parsing algebraic word problems into equations. *Transactions of the Association for Computational Linguistics*, 3:585–597, 2015.

K. Kukich. Design of a knowledge-based report generator. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, pages 145–150. Association for Computational Linguistics, 1983.

N. Kushman, Y. Artzi, L. Zettlemoyer, and R. Barzilay. Learning to automatically solve algebra word problems. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 271–281, 2014.

K. H. Lai, M. Topaz, F. R. Goss, and L. Zhou. Automated misspelling detection and correction in clinical free-text records. *Journal of biomedical informatics*, 55:188–195, 2015.

V. Lampos and N. Cristianini. Tracking the flu pandemic by monitoring the social web. In *Cognitive Information Processing (CIP), 2010 2nd International Workshop on*, pages 411–416. IEEE, 2010.

V. Lampos, D. Preoţiuc-Pietro, and T. Cohn. A user-centric model of voting intention from social media. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 993–1003, 2013.

J. H. Lau, L. Chi, K.-N. Tran, and T. Cohn. End-to-end network for twitter geolocation prediction and hashing. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 744–753, 2017.

C. Leacock, M. Chodorow, M. Gamon, and J. Tetreault. Automated grammatical error detection for language learners. *Synthesis lectures on human language technologies*, 3(1):1–134, 2010.

C. Leacock, M. Chodorow, M. Gamon, and J. Tetreault. Automated grammatical error detection for language learners. *Synthesis lectures on human language technologies*, 7(1):1–170, 2014.

R. Lebret, D. Grangier, and M. Auli. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771*, 2016.

I. Lev, B. MacCartney, C. D. Manning, and R. Levy. Solving logic puzzles: From robust processing to precise semantics. In *Proceedings of the 2nd Workshop on Text Meaning and Interpretation*, pages 9–16. Association for Computational Linguistics, 2004.

J. Li and G. Hirst. Semantic knowledge in word completion. In *Proceedings of the 7th international ACM SIGACCESS conference on Computers and accessibility*, pages 121–128, 2005.

P. Liang, M. I. Jordan, and D. Klein. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 91–99. Association for Computational Linguistics, 2009.

C.-H. Lin, N.-Y. Wu, W.-S. Lai, and D.-M. Liou. Comparison of a semi-automatic annotation tool and a natural language processing application for the generation

of clinical statement entries. *Journal of the American Medical Informatics Association*, 22:132–142, 2014.

M. Lisby, L. P. Nielsen, and J. Mainz. Errors in the medication process: frequency, type, and potential clinical consequences. *International Journal for Quality in Health Care*, 17(1):15–22, 2005.

P. LoBue and A. Yates. Types of common-sense knowledge needed for recognizing textual entailment. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 329–334. Association for Computational Linguistics, 2011.

C. Lovis, R. H. Baud, and P. Planche. Power of expression in the electronic patient record: structured data or narrative text? *International Journal of Medical Informatics*, 58:101–110, 2000.

M.-T. Luong and C. D. Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1054–1063, 2016.

T. Luong, M. Kayser, and C. D. Manning. Deep neural language models for machine translation. In *Proceedings of the 19th Conference on Computational Natural Language Learning, CoNLL 2015*, pages 305–309, 2015.

E. Macaulay, G. Cooper, J. Engeset, and A. Naylor. Prospective audit of discharge summary errors. *British journal of surgery*, 83(6):788–790, 1996.

A. Madaan, A. Mittal, G. Ramakrishnan, S. Sarawagi, et al. Numerical relation extraction with minimal supervision. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

B. J. Maron, J. A. Towbin, G. Thiene, C. Antzelevitch, D. Corrado, D. Arnett, A. J. Moss, C. E. Seidman, and J. B. Young. Contemporary definitions and classification of the cardiomyopathies: an american heart association scientific statement

from the council on clinical cardiology, heart failure and transplantation committee; quality of care and outcomes research and functional genomics and translational biology interdisciplinary working groups; and council on epidemiology and prevention. *Circulation*, 113(14):1807–1816, 2006.

J. Matiasek, M. Baroni, and H. Trost. Fasty—a multi-lingual approach to text prediction. In *International Conference on Computers for Handicapped Persons*, pages 243–250. Springer, 2002.

K. F. McCoy, C. A. Pennington, and A. L. Badman. Compansion: From research prototype to practical integration. *Natural Language Engineering*, 4(1):73–95, 1998.

B. McMahan and M. Stone. A bayesian model of grounded color semantics. *Transactions of the Association for Computational Linguistics*, 3:103–115, 2015.

S. Merity, C. Xiong, J. Bradbury, and R. Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

S. M. Meystre, G. K. Savova, K. C. Kipper-Schuler, and J. F. Hurdle. Extracting information from textual documents in the electronic health record: a review of recent research. *Yearbook of medical informatics*, 17(01):128–144, 2008.

T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association*, pages 1045–1048, 2010.

J. Mitchell and M. Lapata. Language models based on semantic composition. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 430–439. Association for Computational Linguistics, 2009.

A. Mitra and C. Baral. Learning to use formulas to solve simple arithmetic problems. In *ACL*, 2016.

Y. Miyamoto and K. Cho. Gated word-character recurrent language model. *arXiv preprint arXiv:1606.01700*, 2016.

F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005*, 2005a.

F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252. Citeseer, 2005b.

R. Nallapati, B. Zhou, C. dos Santos, C. Gulcehre, and B. Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, 2016.

K. Narisawa, Y. Watanabe, J. Mizuno, N. Okazaki, and K. Inui. Is a 204 cm man tall or small? acquisition of numerical common sense from the web. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 382–391, 2013.

A. Newell, S. Langer, and M. Hickey. The rôle of natural language processing in alternative and augmentative communication. *Natural Language Engineering*, 4 (1):1–16, 1998.

H. T. Ng, S. M. Wu, Y. Wu, C. Hadiwinoto, and J. Tetreault. The CoNLL-2013 Shared Task on Grammatical Error Correction. In H. T. Ng, J. Tetreault, S. M. Wu, Y. Wu, and C. Hadiwinoto, editors, *Proceedings of the CoNLL: Shared Task*, pages 1–12, 2013. URL `http://www.aclweb.org/anthology/W13-3601`.

H. T. Ng, S. M. Wu, T. Briscoe, C. Hadiwinoto, R. H. Susanto, and C. Bryant. The CoNLL-2014 on Grammatical Error Correction. In *CoNLL Shared Task*, pages 1–14, 2014.

D. Nguyen, N. A. Smith, and C. P. Rosé. Author age prediction from text using linear regression. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*, pages 115–123. Association for Computational Linguistics, 2011.

T.-V. T. Nguyen and A. Moschitti. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 277–282. Association for Computational Linguistics, 2011.

J. Novikova, O. Dušek, and V. Rieser. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 201–206, 2017.

D. Paperno, G. Kruszewski, A. Lazaridou, N. Q. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernandez. The lambada dataset: Word prediction requiring a broad discourse context. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1525–1534, 2016.

T. Park, E. Lank, P. Poupart, and M. Terry. Is the sky pure today? awkchecker: an assistive tool for detecting and correcting collocation errors. In *Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 121–130. ACM, 2008.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

M. A. Perryman, L. Lindegren, J. Kovalevsky, E. Hoeg, U. Bastian, P. Bernacca, M. Crézé, F. Donati, M. Grenon, M. Grewing, et al. The hipparcos catalogue. *Astronomy and Astrophysics*, 323:L49–L52, 1997.

R. Prabhavalkar, K. Rao, T. N. Sainath, B. Li, L. Johnson, and N. Jaitly. A comparison of sequence-to-sequence models for speech recognition. In *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association*, pages 939–943, 2017.

J. Prager, J. Chu-Carroll, K. Czuba, C. Welty, A. Ittycheriah, and R. Mahindru. Ibm's piquant in trec2003. Technical report, IBM THOMAS J WATSON RESEARCH CENTER YORKTOWN HEIGHTS NY, 2003.

C. Quirk, C. Brockett, and B. Dolan. Monolingual machine translation for paraphrase generation. 2004.

A. Rahimi, T. Cohn, and T. Baldwin. Twitter user geolocation using a unified text and network prediction model. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 630–636, 2015a.

A. Rahimi, D. Vu, T. Cohn, and T. Baldwin. Exploiting text and network context for geolocation of social media users. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1362–1367, 2015b.

A. Rahimi, T. Baldwin, and T. Cohn. Continuous representation of location for geolocation and lexical dialectology using mixture density networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 167–176, 2017a.

A. Rahimi, T. Cohn, and T. Baldwin. A neural model for user geolocation and lexical dialectology. In *Proceedings of the 55th Annual Meeting of the Associ-*

ation for Computational Linguistics (Volume 2: Short Papers), volume 2, pages 209–216, 2017b.

G. Randhawa, M. Ferreyra, R. Ahmed, O. Ezzat, and K. Pottie. Using machine translation in clinical practice. *Canadian Family Physician*, 59(4):382–383, 2013.

H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*, pages 2931–2937, 2017.

M. Rei and H. Yannakoudakis. Auxiliary objectives for neural error detection models. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, BEA@EMNLP 2017*, pages 33–43, 2017.

E. Reiter and R. Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.

E. Reiter and R. Dale. *Building natural language generation systems*. Cambridge university press, 2000.

E. Reiter, S. Sripada, J. Hunter, J. Yu, and I. Davy. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1-2):137–169, 2005.

B. Roark. Probabilistic top-down parsing and language modeling. *Computational linguistics*, 27(2):249–276, 2001.

S. T. Rosenbloom, J. C. Denny, H. Xu, N. Lorenzi, W. W. Stead, and K. B. Johnson. Data from clinical notes: a perspective on the tension between structure and flexible documentation. *Journal of the American Medical Informatics Association*, 18(2):181–186, 2011.

S. Roy and D. Roth. Solving general arithmetic word problems. *arXiv preprint arXiv:1608.01413*, 2016.

S. Roy, T. Vieira, and D. Roth. Reasoning about quantities in natural language. *Transactions of the Association for Computational Linguistics*, 3:1–13, 2015.

A. Rozovskaya and D. Roth. Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 961–970. Association for Computational Linguistics, 2010a.

A. Rozovskaya and D. Roth. Annotating esl errors: Challenges and rewards. In *Proceedings of the NAACL HLT 2010 fifth workshop on innovative use of NLP for building educational applications*, pages 28–36. Association for Computational Linguistics, 2010b.

J. Ruppenhofer, J. Brandes, P. Steiner, and M. Wiegand. Ordering adverbs by their scaling effect on adjective intensity. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 545–554, 2015.

M. Sammons, V. Vydiswaran, and D. Roth. Ask not what textual entailment can do for you... In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1199–1208. Association for Computational Linguistics, 2010.

C. D. Santos and B. Zadrozny. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826, 2014.

E. Sapir. Grading, a study in semantics. *Philosophy of science*, 11(2):93–116, 1944.

G. K. Savova, J. J. Masanz, P. V. Ogren, J. Zheng, S. Sohn, K. C. Kipper-Schuler, and C. G. Chute. Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

J. Schler, M. Koppel, S. Argamon, and J. W. Pennebaker. Effects of age and gender

on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205, 2006.

R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1715–1725, 2016.

M. Seo, H. Hajishirzi, A. Farhadi, O. Etzioni, and C. Malcolm. Solving geometry problems: Combining text and diagram interpretation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1466–1476, 2015.

M. Sevenster and Z. Aleksovski. Snomed ct saves keystrokes: quantifying semantic autocompletion. In *Proceedings of American Medical Informatics Association (AMIA) Annual Symposium*, 2010.

M. Sevenster, R. van Ommering, and Y. Qian. Algorithmic and user study of an autocompletion algorithm on a large medical vocabulary. *Journal of biomedical informatics*, 45(1):107–119, 2012.

C.-C. Shei and H. Pain. An esl writer's collocational aid. *Computer Assisted Language Learning*, 13(2):167–182, 2000.

S. Shi, Y. Wang, C.-Y. Lin, X. Liu, and Y. Rui. Automatically solving number word problems by semantic parsing and reasoning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Lisbon, Portugal*, 2015.

C. Shivade, M.-C. de Marneffe, E. Fosler-Lussier, and A. M. Lai. Corpus-based discovery of semantic intensity scales. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–493, 2015.

C. Shivade, M.-C. de Marneffe, E. Fosler-Lussier, and A. M. Lai. Identification, characterization, and grounding of gradable terms in clinical text. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 17–26, 2016.

C. Silberer and M. Lapata. Learning Grounded Meaning Representations with Autoencoders. In *Proceedings of ACL*, 2014. URL `http://www.aclweb.org/anthology/P14-1068`.

H. Singh, S. Mani, D. Espadas, N. Petersen, V. Franklin, and L. A. Petersen. Prescription errors and outcomes related to inconsistent information transmitted through computerized order entry: a prospective study. *Archives of internal medicine*, 169(10):982–989, 2009.

R. Sirel. Dynamic user interfaces for synchronous encoding and linguistic uniforming of textual clinical data. In *Human Language Technologies–The Baltic Perspective: Proceedings of the 5th International Conference Baltic HLT*, 2012.

E. Soysal, J. Wang, M. Jiang, Y. Wu, S. Pakhomov, H. Liu, and H. Xu. Clamp–a toolkit for efficiently building customized clinical natural language processing pipelines. *Journal of the American Medical Informatics Association*, 25(3):331–336, 2017.

G. P. Spithourakis and S. Riedel. Numeracy for language models: Evaluating and improving their ability to predict numbers. In *Proceedings of ACL*, 2018.

G. P. Spithourakis, S. E. Petersen, and S. Riedel. Harnessing the predictive power of clinical narrative to resolve inconsistencies and omissions in ehrs. In *Proceedings of NIPS - 2nd Workshop on Machine Learning for Clinical Data Analysis, Healthcare and Genomics*, 2014.

G. P. Spithourakis, I. Augenstein, and S. Riedel. Numerically grounded language models for semantic error correction. In *Proceedings of EMNLP*, 2016a.

G. P. Spithourakis, S. E. Petersen, and S. Riedel. Clinical text prediction with numerically grounded conditional language models. In *Proceedings of EMNLP - LOUHI workshop*, 2016b.

S. Sripada, E. Reiter, and I. Davy. Sumtime-mousam: Configurable marine weather forecast generator. *Expert Update*, 6(3):4–10, 2003a.

S. G. Sripada, E. Reiter, J. Hunter, and J. Yu. Summarizing neonatal time series data. In *Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics-Volume 2*, pages 167–170. Association for Computational Linguistics, 2003b.

N. Srivastava. Improving neural networks with dropout. *University of Toronto*, 182, 2013.

H. Stamerjohanns and M. Kohlhase. Transforming the arχiv to xml. In *International Conference on Intelligent Computer Mathematics*, pages 574–582. Springer, 2008.

H. Stamerjohanns, M. Kohlhase, D. Ginev, C. David, and B. Miller. Transforming large collections of scientific publications to xml. *Mathematics in Computer Science*, 3(3):299–307, 2010.

M. H. Stanfill, M. Williams, S. H. Fenton, R. A. Jenders, and W. R. Hersh. A systematic literature review of automated clinical coding and classification systems. *Journal of the American Medical Informatics Association*, 17(6):646–651, 2010.

I. Sutskever, J. Martens, and G. E. Hinton. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, 2011.

I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

B. Tang, Y. Wu, M. Jiang, Y. Chen, J. C. Denny, and H. Xu. A hybrid system for temporal information extraction from clinical text. *Journal of the American Medical Informatics Association*, 20(5):828–835, 2013.

J. Tetreault, M. Chodorow, and N. Madnani. Bucking the trend: improved evaluation and annotation practices for esl error detection systems. *Language Resources and Evaluation*, 48(1):5–31, 2014.

J. R. Tetreault and M. Chodorow. The ups and downs of preposition error detection in esl writing. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 865–872. Association for Computational Linguistics, 2008.

K. Trnka. Adaptive language modeling for word prediction. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, 2008.

K. Trnka and K. F. McCoy. Corpus studies in word prediction. In *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, pages 195–202, 2007.

K. Trnka and K. F. McCoy. Evaluating word prediction: framing keystroke savings. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, 2008.

J. Tzelgov, D. Ganor-Stern, A. Y. Kallai, and M. Pinhas. Primitives and non-primitives of numerical representations. *Oxford library of psychology. The Oxford handbook of numerical cognition*, pages 45–66, 2015.

J. Ueberla. Analysing a simple language model· some general conclusions for language models for speech recognition. *Computer Speech & Language*, 8(2):153–176, 1994.

S. Upadhyay and M.-W. Chang. Annotating derivations: A new evaluation strategy and dataset for algebra word problems. *arXiv preprint arXiv:1609.07197*, 2016.

S. Upadhyay, M.-W. Chang, K.-W. Chang, and W.-t. Yih. Learning from explicit and implicit supervision jointly for algebra word problems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 297–306, 2016.

Ö. Uzuner. Recognizing obesity and comorbidities in sparse data. *Journal of the American Medical Informatics Association*, 16(4):561–570, 2009.

Ö. Uzuner, I. Goldstein, Y. Luo, and I. Kohane. Identifying patient smoking status from medical discharge records. *Journal of the American Medical Informatics Association*, 15(1):14–24, 2008.

Ö. Uzuner, I. Solti, and E. Cadag. Extracting medication information from clinical text. *Journal of the American Medical Informatics Association*, 17(5):514–518, 2010.

A. Van Den Bosch and T. Bogers. Efficient context-sensitive word completion for mobile devices. In *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, 2008.

E. M. Vecchi, M. Baroni, and R. Zamparelli. (Linear) Maps of the Impossible: Capturing semantic anomalies in distributional space. In *Proceedings of the Workshop on Distributional Semantics and Compositionality*, pages 1–9, 2011.

O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015a.

O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015b.

T. Wandmacher and J.-Y. Antoine. Methods to integrate a language model with semantic information for a word prediction component. In *Proceedings of EMNLP*, 2008.

T. Wandmacher, J.-Y. Antoine, F. Poirier, and J.-P. Départe. Sibylle, an assistive communication system adapting to the context and its user. *ACM Transactions on Accessible Computing (TACCESS)*, 1(1):6, 2008.

T. Wang, X. Yuan, and A. Trischler. A joint model for question answering and question generation. *CoRR*, abs/1706.01450, 2017.

Y. Wang, L. Wang, M. Rastegar-Mojarad, S. Moon, F. Shen, N. Afzal, S. Liu, Y. Zeng, S. Mehrabi, S. Sohn, et al. Clinical information extraction applications: a literature review. *Journal of biomedical informatics*, 77:34–49, 2018.

C. R. Weir, J. Hurdle, M. Felgar, J. Hoffman, B. Roth, and J. Nebeker. Direct text entry in electronic progress notes. *Methods of information in medicine*, 42(01): 61–67, 2003.

D. Wible, C.-H. Kuo, N.-L. Tsao, A. Liu, and H.-L. Lin. Bootstrapping in a language learning environment. *Journal of Computer Assisted Learning*, 19(1):90–102, 2003.

S. Wiseman, S. Shieber, and A. Rush. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, 2017.

M. E. Wood and E. Lewis. Windmill-the use of a parsing algorithm to produce predictions for disabled persons. *PROCEEDINGS-INSTITUTE OF ACOUSTICS*, 18:315–322, 1996.

J.-C. Wu, Y.-C. Chang, T. Mitamura, and J. S. Chang. Automatic collocation suggestion in academic writing. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 115–119. Association for Computational Linguistics, 2010.

W. Xiong, J. Droppo, X. Huang, F. Seide, M. Seltzer, A. Stolcke, D. Yu, and G. Zweig. The microsoft 2016 conversational speech recognition system. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017*, pages 5255–5259, 2017.

L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

X. Yi, J. Gao, and B. Dolan. A web-based english proofing system for english as a second language users. 2008.

D. Yogatama, M. Heilman, B. O'Connor, C. Dyer, B. R. Routledge, and N. A. Smith. Predicting a scientific community's response to an article. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 594–604. Association for Computational Linguistics, 2011.

M. Yoshida, I. Sato, H. Nakagawa, and A. Terada. Mining numbers in text using suffix arrays and clustering based on dirichlet process mixture models. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 230–237. Springer, 2010.

J. Yu, E. Reiter, J. Hunter, and C. Mellish. Choosing the content of textual summaries of large time-series data sets. *Natural Language Engineering*, 13(1):25–49, 2007.

M. D. Zeiler. Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

Y. Zhang, W. Chan, and N. Jaitly. Very deep convolutional networks for end-to-end speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2017*, pages 4845–4849, 2017.

L. Zhou, S. Dai, and L. Chen. Learn to solve algebra word problems using quadratic programming. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (Lisbon, Portugal*, pages 817–822, 2015.