TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

JAAKKO PASANEN

NATURAL LANGUAGE UNDERSTANDING FOR INFORMATION RETRIEVAL IN CHAT BASED CUSTOMER SUPPORT

Master of Science thesis

Examiner: Prof. Ari Visa
Examiner and topic approved by the
Faculty Council of the Faculty of
Engineering Sciences
on 31st December 2016

ABSTRACT

JAAKKO PASANEN: Natural Language Understanding for Information Retrieval in Chat Based Customer Support
Tampere University of Technology
Master of Science thesis, xx pages, x Appendix pages
December 2016
Master's Degree Programme in Automation Technology
Major: Learning and Intelligent Systems
Examiner: Prof. Ari Visa
Keywords: Hype

The abstract is a concise 1-page description of the work: what was the problem, what was done, and what are the results. Do not include charts or tables in the abstract.

Put the abstract in the primary language of your thesis first and then the translation (when that is needed).

TIIVISTELMÄ

JAAKKO PASANEN: Luonnollisen kielen ymmärrys tiedon hakua varten chat-pohjaisessa asiakaspalvelussa
Tampereen teknillinen yliopisto
Diplomityö, xx sivua, x liitesivua
Joulukuu 2016
Automaatiotekniikan koulutusohjelma
Pääaine: Oppivat ja älykkäät järjestelmät
Tarkastajat: Prof. Ari Visa
Avainsanat: Hype

The abstract in Finnish. Foreign students do not need this page.

Suomenkieliseen diplomityöhön kirjoitetaan tiivistelmä sekä suomeksi että englanniksi.

Kandidaatintyön tiivistelmä kirjoitetaan ainoastaan kerran, samalla kielellä kuin työ. Kuitenkin myös suomenkielisillä kandidaatintöillä pitää olla englanninkielinen otsikko arkistointia varten.

PREFACE

This document template conforms to Guide to Writing a Thesis at Tampere University of Technology (2014) and is based on the previous template. The main purpose is to show how the theses are formatted using LaTeX (or LaTeX to be extra fancy) .

The thesis text is written into file `d_tyo.tex`, whereas `tutthesis.cls` contains the formatting instructions. Both files include lots of comments (start with %) that should help in using LaTeX. TUT specific formatting is done by additional settings on top of the original `report.cls` class file. This example needs few additional files: TUT logo, example figure, example code, as well as example bibliography and its formatting (`.bst`) An example makefile is provided for those preferring command line. You are encouraged to comment your work and to keep the length of lines moderate, e.g. <80 characters. In Emacs, you can use `Alt-Q` to break long lines in a paragraph and `Tab` to indent commands (e.g. inside figure and table environments). Moreover, tex files are well suited for versioning systems, such as Subversion or Git.

Acknowledgements to those who contributed to the thesis are generally presented in the preface. It is not appropriate to criticize anyone in the preface, even though the preface will not affect your grade. The preface must fit on one page. Add the date, after which you have not made any revisions to the text, at the end of the preface.

Tampere, 11.8.2014

On behalf of the working group, Erno Salminen

CONTENTS

LIST OF ABBREVIATIONS AND SYMBOLS

ANN	Artificial Neural Network
LAS	Labelled Attachment Score
LDA	Latent Dirilecht Allocation
LSA	Latent Semantic Analysis
LSTM	Long short term memory; type of RNN with short term memory.
NER	Named entity recognition
NLP	Natural Language Processing
PMI	Pointwise mutual information
POS	Part-of-speech; also called lexical category
RNN	Recurrent neural network
S-LSTM	Stack long short term memory
TUT	Tampere University of Technology
UAS	Unlabelled Attachment Score

NOTES

0.1 Reading

These vectors can be used as features in a variety of applications, such as information retrieval (Manning et al., 2008), document classification (Sebastiani, 2002), question answering (Tellex et al., 2003). Pennington et al. 2014

See first paragraph of section 1 of Liang et al. 2016 for sources on machine translation

Encoder-decoder model

0.2 Terms for Computational Linguistics

1-of-V Coding Representing words as sparse binary vectors which have 1 at the word's vocabulary index and 0 all others. With vocabulary {dog, cat, mouse}, dog becomes [1, 0, 0], cat becomes [0, 1, 0] and mouse [0, 0, 1].

Bag-of-words Multiset of words appearing in a text with occurrence counts for each word. Used as a tool for feature generation. Does not preserve word order or grammar. Can implemented as a dictionary (or associative array) where words are the keys and counts are the values.

Conditional Random Field

Constituent In syntactic analysis, a constituent is a word or a group of words that function(s) as a single unit within a hierarchical structure. Many constituents are phrases. *Yesterday I saw* ***an orange bird with a white neck***

Corpus A collection of texts with linguistic annotations.

Dimensionality When discussing word embeddings and word vector spaces the dimensionality refers to definition in linear algebra. Dimensionality of arrays in computing means the number of indices required to specify an element in the array. Word vector in 50 dimensional vector space $\mathbb{R}^{50}$ would be represented in computing as one dimensional array of length 50 [d1, d2, d3, ..., d50]

Distributional Hypothesis Words that are used and occur in the same contexts tend to purport similar meanings. Harris 1954

Feature Numeric data representation that can be effectively exploited in machine learning tasks. E.g. Word occurrence frequencies.

Feature Vector Vector containing all the features. For an image a feature vector could be all the raw values of pixels as a single sequence. For a trigram model with 300 dimensional word embeddings a feature vector would be a 900 dimensional vector formed by concatenating all the separate word embedding vectors.

Gazzetteer In Named Entity Recognition a gazzetteer is a dictionary of known named entities.

Language Model Probability distribution over sequences of words. Given such a sequence, say of length m, it assigns a probability $P(w_1, \dots, w_m)$ to the whole sequence. Problems caused by growing vocabulary can be addressed with continuous language models such as neural net language models (NNML). Word2Vec by Mikolov, Corrado, et al. 2013 addresses this problem with Continuous Bag-of-words and Skip-gram models.

Lemmatisation Process of finding the base form of a word, e.g. flew -> fly

Lexeme A basic lexical unit of a language consisting of one word or several words, the elements of which do not separately convey the meaning of the whole.

n-gram Probabilistic language model where probability of current word is the joint probability of previous n words. Bigram example: $P(I, saw, the, red, house) \approx P(I|\hat{}\,)P(saw|I)P(red|the)P(house|red)P(\$|house)$. The words unigram, bigram and trigram language model denote n-gram model language models with n = 1, n = 2 and n = 3, respectively.

One-hot Group of bits which the legal comibations of values are only those with a single high (1) and all the others low (0). See also 1-of-V Coding.

Parsing Within computational linguistics the term is used to refer to the formal analysis by a computer of a sentence or other string of words into its constituents, resulting in a parse tree showing their syntactic relation to each other, which may also contain semantic and other information.

PMI Word co-occurence probability metric. High values for words that occur often together.

POS-tagging Process of marking up a word to particular part-of-speech (nouns, verbs, etc...) based on both its definition and its context.

ReLU Rectified Linear Unit. $h(x) = max\{0, x\}$. Used as non-linear activation function in neural nets particularly in convolutional net

Skip-gram Language model which predicts the context (previous and next *n* words) of a current word from the current word instead of traditional way of predicting current word from the context.

Structured Prediction Predicting structured objects, rather than scalar discrete or real values. Translating a natural language sentence into a syntactic representation such as a parse tree can be seen as a structured prediction problem in which the structured output domain is the set of all possible parse trees.

Token A structure representing a lexeme that explicitly indicates its categorization for the purpose of parsing. In plain words tokens are instances of words in a text. Not to be confused with word type.

Tree bank Parsed text corpus that annotates syntactic or semantic sentence structure. Contains trees for sentences where phrases in a sentence are structured in a tree of syntactic or semantic relations. Very useful for training POS-taggers etc...

Tri-Training Parsing unlabeled data with two different parses and selecting only the sentences for which the two parsers produce the same trees Weiss et al. 2015

Word Lookup Table Matrix $\mathbf{P} \in \mathbb{R}^{d \times |V|}$ of d rows and |V| columns, where d is the word vector dimensionality and |V| is the size of vocalbulary. I.e. each column represent a single word and each row represent single dimension in vector space.

Word Type Unique words in a text. *Good wine is good* has 4 tokens but only 3 word types.

Word vector N-dimensional vector representation of a word with interesting properties such as: vector('Paris') - vector('France') + Vector('Italy') -> vector('Rome')

0.3 Universal Dependecies

0.3.1 CoNNL-U format

Universal dependencies use CoNNL-U format for treebanks, CoNNL-U is revised version of CoNNL-X. Annotations are encoded in text files with word lines, blank lines for sentence boundaries and comments starting with hash (#).

Word lines consist of following columns:

ID	Word ID in sentence
FORM	Word form or punctuation symbol
LEMMA	Lemma or stem of word form
UPOSTAG	Universal part-of-speech tag
XPOSTAG	Language specific part-of-speech tag
FEATS	List of morphological features
HEAD	Head of the curren token, value of ID or zero (0)
DEPREL	Universal dependecy relation to the HEAD
DEPS	List of secondary dependencies
MISC	Any other annotation

Example in Finnish: Jäällä kävely avaa aina hauskoja ja erikoisia näkökulmia kaupunkiin

ID	FORM	LEMMA	UPOSTAG	XPOSTAG
1	Jäällä	jää	NOUN	N
2	kävely	kävely	NOUN	N
3	avaa	avata	VERB	V
4	aina	aina	ADV	Adv
5	hauskoja	hauska	ADJ	A
6	ja	ja	CONJ	C
7	erikoisia	erikoinen	ADJ	A
8	näkökulmia	näkö#kulma	NOUN	N
9	kaupunkiin	kaupunki	NOUN	N
10	.	.	PUNCT	Punct

FEATS
Case=Ade|Number=Sing
Case=Nom|Number=Sing
Mood=Ind|Number=Sing|Person=3|Tense=Pres|VerbForm=Fin|Voice=Act
—
Case=Par|Degree=Pos|Number=Plur
—
Case=Par|Degree=Pos|Number=Plur
Case=Par|Number=Plur
Case=Ill|Number=Sing
—

HEAD	DEPREL	DEPS	MISC
2	nmod	_	_
3	nsubj	_	_
0	root	_	_
3	advmod	_	_
8	amod	_	_
5	cc	_	_
5	conj	8:amod	_
3	dobj	_	_
8	nmod	_	SpaceAfter=No

0.3.2 Universal POS tags

ADJ	Adjective. Describing word qualifying noun or noun phrase. **deep**, **intelligent**
ADP	Adposition. Word expressing spatial or temporal relations **under**, **around**, **before** or mark various semantic roles **of**, **for**
ADV	Adverb. Modifies another word. Typically express manner, place, time, frequency etc. She sang **loudly**. You are **quite** right.
AUX	Auxiliary verb. A verb used in forming the tenses, moods, and voices of other verbs. **Do** you want tea?. He **has** given his all.
CONJ	Coordinating conjunction. Conjunction placed between words, phrases, clauses or sentences of equal rank. **and**, **but**, **or**.
DET	Determiner. Expresses reference of a noun (group). **The** girl is **a** student. **Which** book is that?
INTJ	Interjection. Shows emotion or feeling of the author, includes exclamations, curses, greetings and such. **Ouch!**, **hey**, **huh?**.
NOUN	Noun. Denotes a person, animal, place thing or idea. The **cat** sat on a **mat**.
NUM	Numeral. Number, written with digits or letters. **12**, **eleven**.
PART	Particle. Cannot be inflected. Interjections and conjunctions. In finnish also **että**, **jotta**, **koska**, **kun** etc...
PRON	Pronoun. Replaces (often previously introduced) noun. Joe saw Jill, and **he** waved at **her**.
PUNCT	Punctuation. Full stop, comma, bracket etc.
SCONJ	Subordinating conjunction. A conjunction that introduces a subordinating clause, e.g. **although**, **because**, **whenever**.
SYM	Symbol.
VERB	Verb. Conveys an action **bring**, **read**, an occurrence **happen**, **become**, or a state of being **be**, **exist**.

X	Other

1. INTRODUCTION

Testing citation Andor et al. 2016

2. NATURAL LANGUAGE PROCESSING

- Natural Language Processing is vastly wide field, this thesis discusses only on the sections of NLP relevant to the experiments.
- Subfields such as sentence segmentation and sentiment analysis are out of scope of this thesis.
- Most of the NLP work has been for english.
- Cross-linguistic annotation and parsing has been a reality only after introduction of The Universal Dependencies project and SyntaxNet.
- Similarly Finnish parsing has been unreachable until the first Finnish corpus Turku Dependency Treebank Haverinen et al. 2014 and cross-linguistic parsers.
- Traditionally NLP systems are tailored to the single problem at hand with hand engineered features suited for the problem. Recently general approach has received interest where feature engineering and task specific architectures are not needed. Collobert et al. 2011, Zhang and LeCun 2015
- This thesis focuses on task specific systems because general language understanding systems are still lacking a good support for production systems and depend on vast amounts of good data Zhang and LeCun 2015 unobtainable within our time frame and resources.
- See Chapter 2.1 for Finnish Language quirks in Korenius et al. 2004

2.1 Pre-processing

2.2 Feature Engingeering in NLP

- Machine learning algorithms require words to be represented quantifiable features such as IDs or real number vectors.
- Traditional feature selection requires hand engineered features.

- Engineered features hog 95% of the computation time. Chen and Manning 2014
- Traditionally words have been represented by indices. Mikolov, Corrado, et al. 2013
- Next step was to use 1-of-V coding.
- Index representation is simple as computationally cheap, making use of huge datasets possible. Simple models with huge data outperform complex models with less data. Mikolov, Corrado, et al. 2013
- See LSA and LDA for previous systems. Neural networks significantly outperform LSA in preserving linearities. LDA doesn't scale for large datasets. Mikolov, Corrado, et al. 2013

2.2.1 Word Embeddings

- **see section 1.2 of Mikolov, Corrado, et al. 2013 for previous work and history of word embeddings**
- Word embeddings represent words as n-dimensional vectors. Mikolov, Corrado, et al. 2013
- LSA leverages statistical information of a corpus but performs poorly on word analogy task. Pennington et al. 2014
- Skip-gram is good for word analogies but doesn't utilize corpus statistics well since vectors are trained on local context. Pennington et al. 2014
- Word embeddings try to map words with semantic similarities close to each other. Words may have several types of similarities such as *France* and *Italy* are countries but *dogs* and *triangles* are both in plural form. Mikolov, Yih, et al. 2013
- Chen and Manning 2014 use 50 dimensional word embeddings created with Word2vec.
- Chen and Manning 2014 also use embeddings for POS tags and dependecy arcs. Only embedding POS tags has clear benefit, Chen and Manning 2014 suspect that embedding arc labels have no effect since POS tags already contain the relational information.

- Word embeddings with lookup table generalize poorly with morpohlogically rich languages such as Finnish. Takala 2016
- Morphologically rich languages benefit from breaking the word into sub-parts, RNN based character level model is not compared with Stem+ending. Takala 2016
- Word embeddings obtained through neural language models exhibit the property whereby semantically close words are close in the embedding vector space. Kim et al. 2016

2.2.2 Word2vec

- Mikolov, Corrado, et al. 2013
- Can be used with datasets of billions of words
- Has two models: Continuous bag-of-words and continuous skip-gram
- Continuous bag-of-words predicts current word from the context (surrounding words)
- Continuous skip-gram predicts context (surrounding words) from current word.
- Continuous Bag-of-Words is better for small datasets, continuous skip-gram is better for large datasets.
- CBOW is better for syntax, Skip-gram is better for semantics.
- Can be used to find semantic relationships like vector('biggest') - vector('big') + vector('small') => vector('smallest')
- State of the art (as of 2013)

2.2.3 GloVe

- GloVe by Pennington et al. 2014 capture global corpus statistics with log-bilinear co-occurence count model.
- Memory requirements for GloVe are substantial since global co-occurence matrix for entire vocabulary is required, even though GloVe eliminates the need for zero occurence elements. Problem becomes worse for inflectional languages such as Finnish were vocabulary requires word type for each infliction for each word.
- GloVe outperforms other methods on almost all tested tasks. All tasks are English only. Pennington et al. 2014

2.2.4 Charater to Word

- Ling et al. 2015
- Word embeddings can be generated from character sequences with significantly better performance for morphological languages.
- Requires only single vector for each character type. Particularly good for morphological languages where word type count may be infinite.
- Orthographical and functional (syntactic and semantic) relationships are non-trivial: *butter* and *batter* are orthographically close but functionally distant, *rich* and *affluent* are orthographically distant but functionally close. Ling et al. 2015 resort to LSTM networks for learning the relationships.
- Word lookup tables are unable to generate representations for previously unseen words, as is required for morphology. Ling et al. 2015
- C2W can generate embeddings for unseen words.
- C2W is computationally more expensive than word lookup tables, but can be eased by saving word embeddings for most frequent words since the words embedding for a character sequence (word) does not change.
- During training word embeddings change but not inside a single batch, thus it is computationally cheaper to use large batches for training.
- C2W can be replaced with word lookup tables for downstream processing since input and output of both methods are the same.

2.3 Annotations

- Stanford Dependencies by De Marneffe et al. 2006
- Stanford Dependencies emerged as de facto annotation scheme for english, but has been adapted to several other languages including Finnish. Nivre et al. 2016, Haverinen et al. 2014.
- Turku Dependency Treebank has been tranformed into universal dependencies. Pyysalo et al. 2015
- Unified annotation scheme reduces need for cross-language adaptations in downstream development. Petrov et al. 2012

- Universal Dependencies project started from the requirement for cross-linguistically consistent treebank annotations even for morphological languages. Nivre et al. 2016.
- Universal Dependencies project was born from merging several previous attempts to form a cross-linguistically sound dependency annotation schemes. Nivre et al. 2016
- UD data has been encoded in the CoNLL-U format, a revision of the popular CoNLL-X format. Nivre et al. 2016
- UD treebanks released in November 2015. Nivre et al. 2016

2.3.1 Turku Dependecy Treebank

- Haverinen et al. 2014
- Treebanks are needed in computational linguistics.
- First Finnish treebank.
- Open licence, including for text annotated
- 204339 tokens, 15126 sentences
- Based on Stanford Dependency scheme with minor modifications to exclude phenomena not present in Finnish and to include new annotations not present in English.
- Transposed to CoNNL-U scheme by universal dependencies project
- Connexor Machinese Syntax is the only currenty available Finnish full dependency parser.
- Texts from 10 different categories ranging from news and legal text to blog entries and fiction.
- Dependency parsing is done manually with full double annotation process.
- Uses Omorfi for morphological analysis. Ambiguous tokens are handled partly manually, partly rule based and partly with machine learning.
- FTB uses 3 different taggers for morphology, check them out!
- FTB is 97% grammar examples, meant for rule based POS tagger development

2.4 Syntactic Parsing

- Commonly divided to constituency parsing and dependency parsing.
- Constituency parser creates a parse tree of constituencies.
- Dependency parser creates a parse tree of word token dependencies.
- Constituency parsers are slower but more informal than dependency parsers. Fernández-González and Martins 2015
- Fernández-González and Martins 2015 show that it is possible to build constituency parser with dependency parser by reducing constituents to dependency parsing.

2.4.1 POS-tagging

- Started from rule based taggers
- Tagger by Brill 1992 (known as Brill tagger) learns the rules and as such can be considered as a hybrid approach
- Contemporary research is focused on statistical and ANN based taggers
- Rest of this section focuses on statistical parsers
- Ling et al. 2015 introduced S-LSTM based State-of-the-art tagger
- Andor et al. 2016 Improved accuracy with transition based tagger
- Chen and Manning 2014 were first to represent POS-tag and arc labels as embeddings
- Andor et al. 2016 and Weiss et al. 2015 built their solutions based on Chen and Manning 2014
- Nivre 2004 introduced system for transition based taggers known as arc-starndard system. Chen and Manning 2014

2.4.2 Transition Based Parsers

- Good balance between efficiency and accuracy Weiss et al. 2015

- Parsed left to right; at each position the parses chooses action from a set of possible actions.
- Greedy models are fast but error prone and need hand engineered features Weiss et al. 2015
- Actions can be chosen by ANN to avoid hand engineering Chen and Manning 2014, Weiss et al. 2015

2.4.3 Syntaxnet

- Andor et al. 2016
- Transition based
- Locally and globally normalized
- Backpropagation through entire net
- State-of-the-Art

2.5 Language Modeling

- A language model is a probability distribution over a sequence of words, traditionally performed with n-th order Markov assumption or n-gram counting and smoothing (Chen and Goodman, 1998) *see Kim et al. 2016 Introduction for citation.*
- Kim et al. 2016 have character level encoding and word-level predicting model for language modeling.
- Kim et al. 2016 noticed that character inputs are sufficient for language modeling.
- Character level model of Kim et al. 2016 outperfom word-level and morpheme-level models on morphologically rich languages Arabic, Czech, French, German, Spanish and Russian.
- Neural Language Models NLM are blind to sub-word level information (e.g. morphemes). Kim et al. 2016
-

2.6 Machine Translation

See section ?? for more info on our implementation.

2.7 Lemmatisation

- Lemmatization is the process of finding a base form for a word.
- Lemmatization is a normalization technique. Korenius et al. 2004
- Homographic and inflectional word forms cause ambiquity. Korenius et al. 2004
- Compound words cause problems. Korenius et al. 2004
- Lemmatization is better than stemming for clustering of documents written in Finnish because of it's highly inflectional nature. Korenius et al. 2004
- Lemmatization catches better the semantic meaning of a word, as can be deducted from a better clustering performance.
- Omorfi does lemmatization based on morphological analysis
- Omorfi produces multiple lemmas which need to be disambiguated
- Disambiguation can be done with selecting most probable word, given the context, by language model
- Kestemont et al. 2016 try to solve lemmatization as a neural net classification problem, where lemmas are the class labels
- Method of Kestemont et al. 2016 cannot produce lemmas not seen on training time. This restriction also applies to Word2vec by Mikolov, Corrado, et al. 2013
- Lemmatization has received a lot of research attention for highly inflectional languages, see **Kestermont2016**
- Unknown token can be taken as a variation of known token, search with e.g. Levenshtein edit distance
- There has been almost none previous work using deep learning for lemmatization before Kestemont et al. 2016

2.8 Machine Translation

- See section 2 of Kestemont et al. 2016 for sources on Internet to standard language translitteration
- NLP tools suffer with texts with a lot of ortographical variation, one solution is to translate them. Kestemont et al. 2016

2.9 Intent Recognition and Slot Detection

- Intent prediction is determining user's intents from their utterances (messages). Intents can be seen as functions to call in traditional programming.
- Slot detection is identifying relevant actionable pieces of utterance Bhargava et al. 2013. Slots can be seen as function parameters.
- Bhargava et al. 2013 reduce error rates of intent prediction by 6,7% and 8.7% for transcribed data and automatically recognized data respectively when using intents from previous messages.
- Bhargava et al. 2013 find no significant difference for slot detection by using context information.
- Performance of Bhargava et al. 2013 for intent prediction is increased from 97,1% to 97.3% on transcribed data. However they use Viterbi algorithm with full access to future of the dialog, not realistic in production.
- Most systems assume a single intent per utterance leading to unnatural dialogue experience. Xu and Sarikaya 2013
- Approaching multi-intent recognition by selecting top-K hypotheses from a single intent classifier yields poor results. Xu and Sarikaya 2013
- Multi-label learning works better. Splitting into a K binary classifiers, or combining multiple labels into a single label classification problem. Xu and Sarikaya 2013
- Usually K is a system design choice. Xu and Sarikaya 2013

3. DEEP LEARNING

3.1 Activation Functions

- Sigmoid has a problem with vanishing gradients when using multiple hidden layers
- ReLU fixes this problem (still has exploding gradients problem)
- ReLU6 solves expoloding gradients also, but introduce vanishing gradients when x > 6?
- Softplus is the smooth version of ReLU but computationally more expensive.
- ELU is like ReLU but doesn't die off to zero. Clevert et al. 2015

3.2 Recurrent Neural Networks

- DNNs (Deep Neural Networks) cannot be used to map sequences to sequences since they require the dimensionality of inputs and outputs to be fixed. Sutskever et al. 2014
- RNN is suted for medling sequential phenomena. Kim et al. 2016
- In theory RNN can summarize all historical information, but in practice vanilla RNN performs poorly with long sequences due to vanishing/exploding gradients. (Bengio, Simard and Frasconi 1994), *see Model chapter of Kim et al. 2016*
- Long short-term memory (LSTM) networks address the problem of vanishing gradients with long sequences by adding a memory cell. (Hochreiter and Schmidhuber 1997), *see Model chapter of Kim et al. 2016*
- Gradient exploding remains a problem, but is easily addressable in practice by simple strategies such as gradient clipping. Kim et al. 2016

- Adding more layers such that input of a layer is the hidden state of previous layer is often crucial for significant performance increase. (Pascanu et al. 2013), *see Model chapter of Kim et al. 2016*
- Deep LSTM of Sutskever et al. 2014 significantly outperformed their shallow LSTM, each layer decreasing the perplexity by nearly 10%.
- Encoder-decoder used in translation from English to French gains significant performance increase when reversing the source sentence word order. Sutskever et al. 2014
- Sutskever et al. 2014 speculate that reversing the source sentence helps by making backpropagation work better with shorter dependencies of the sentences' first words. Reversing the source sentence did not deteriorate the translation performance of later parts of the sentence, as was initially believed by Sutskever et al. 2014
- Bi-directional RNN is composed of two RNN of which the first reads the sequence in forward direction and the second reads the sequence in reverse direction, hidden states are concatenated. Chung et al. 2016

3.3 Encoder-Decoder

- Introduced by Sutskever et al. 2014 and Cho et al. 2014
- Used in machine translation Chung et al. 2016
- Dual RNN architechture where 1st RNN encodes a sequence of tokens to fixed length vector and 2nd RNN decodeds that vector representation to a target sequence of tokens. Cho et al. 2014, Bahdanau et al. 2014, Sutskever et al. 2014
- Both RNNs are jointly trained to maximize conditional probability of a target sequence given a source sequence. Cho et al. 2014, Bahdanau et al. 2014
- Encoder creates a summary of the entire input sequence. Cho et al. 2014
- Decoder samples a token at a time using input sequence summary, it's own RNN hidden state(s) and/or previously generated sample(s). Cho et al. 2014, Bahdanau et al. 2014
- See figure 1 on page 2 of Cho et al. 2014 for architecture depiction.

- Can be used to generate a target sequence based on input sequence. Can also be used to score a given pair of input-output sequences. Cho et al. 2014
- RNN Encoder-decoder captures semantic and syntactic structures of phrases. Cho et al. 2014
- Encoder-decoder needs to compress all the relevant information of the sentece in a single fixed length vector. This becomes a problem when sentence length increases, larger model is required. Bahdanau et al. 2014.
- Encoder-decoder have no explicit alignment. B. Liu and Lane 2016
- Input and output sequences can be of different length. **Citation?**
- Chung et al. 2016 have character level encoder-decoder sequence-to-sequence machine translation system. Using sub-word level symbols in source side, full character level only in decoder.
- Chung et al. 2016 use novel RNN (bi-scale RNN) on the target side for better handling of multiple timescales
- Character level decoder relaxes the problem with computational complexity (of softmax function) with large target vocabulary. Chung et al. 2016
- Using character level only encoding and decoding removes the need to know how to do segmentation of characters into words, which is a problem in models which use character level word encodings such as C2W. Chung et al. 2016
- All inflectional forms of word result in very large vocabulary, more efficient encoding can achieved with lexeme (lemma) and morphemes, but requires a lemmatizer and morphological analyser. Chung et al. 2016
- Furthermore model may not perform well with common words if the morphological form is rare. Chung et al. 2016

3.4 Attention Mechanism

- Bahdanau et al. 2014 introduced attention mechanism in neural machine translation as a solution to deteriorating performance with long input sentences. Sutskever et al. 2014 speculate that similar improvement could have been gained with simply reversing the source sentence word order.
- System of Bahdanau et al. 2014 soft searches words in source sentence for each word in target sentence.

- System of Bahdanau et al. 2014 does not try to encode the whole sentence as a fixed size vector, but instead input sentence is encoded as sequence of vectors which are weighted at the decoding time.
- See section 3.1 of Bahdanau et al. 2014 for description of decoder with attention.
- Bahdanau et al. 2014 use beam search to approximate maximum conditional probability on the trained model.
- Attention allows encoder-decoder to learn soft alignment of input and output sequences. B. Liu and Lane 2016

4. OUR SYSTEM

- This thesis only considers customer message understanding.
- Greetings, Chatting, information retrieval and response generation are important part of a dialogue system but have been left out of scope of this thesis.
- Large portion of the research on the conversational models focus on chatter systems where utterance responses are retrieved or generated from response corpus. These systems do not aim to solve customer problem by incorporating relevant information retrieved from a specific knowledge base.
- Utterance systems are not suitable for customer support out side of simple frequently asked questions.
- Furthermore large portion of responses generated by neural conversation models are safe responses e.g. *I don't know* Li et al. 2015.
- Conversational models trained on customer support history incorporate data from long time period and as such data may contain outdated information which cannot be corrected unless sentences containing outdated information are replaced. Finding and replacing such information from customer support dialogues requires reading much of the dialogues by humans and rewriting conversation history; task both time consuming, error prone and expensive.
- Also incorporating new information into such dataset requires manual conversation synthesis where humans write new simulated conversations into a dialogue history used for training the system.
- Other ways for updating old and adding new information are required for real world customer support. Our system draws it's information from a knowledge base with traditional user interface for updating and adding new knowledge.

4.1 Pipeline

1. Spell checking and correction (Omorfi)

2. Machine translation from colloquial speech to standard dialect
3. Co-Reference replacement
4. Named Entity Recognition and replacement
5. POS-tagging (Syntaxnet)
6. Dependency parsing (Syntaxnet)
7. Lemmatization (Omorfi)
8. Character to word with Ling et al. 2015 for word lemmas and POS-tags
9. 1-of-V coding for features
10. Dependencies?
11. Query generation
12. Querying knowledge base with generated query
13. Response rendering with templates (sprintf?)

4.2 C2W

- Character vocabulary needs to fixed for character embedding layer.
- We selected !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~ÄÅÖäåö€ as our character vocabulary.
- This contains ASCII characters from 32 to 127, scandic characters in lower and upper case and € sign.
- Selected character vocabulary covers 99,933% of character usages in Finnish internet parsebank n-gram dataset.
- ASCII end-of-text (ETX) and substition (SUB) characters were added to character vocabulary to handle sequence length padding and out of vocabulary characters respectively.

4.3 C2W2V

- Learning word2vec embeddings from character level sequences by mapping character embeddings sequentially with RNN to pre-trained word2vec vector representations.
- Training with word2vec vocabulary of 2208293 words
- Word2vec embeddings trained by Ginter and Kanerva 2014 from 5-gram data.
- character level sequences are embedded with C2W architecture of Ling et al. 2015.
- Using additive inverse of cosine similarity of word2vec embedding and predicted C2W embedding as loss function.
- Using mean squared error as loss function yielded even worse results when validating with cosine similarities of different words and comparing to cosine similarities of the same words calculated from word2vec embeddings.
- Optimizing with Adam gradient descent optimizer.
- Training converged to cosine similarity of 0.3680 and would not get past that with varying hyperparameters and increasing model size.
- Fail to learn word2vec representations from character level sequences is probably due to word2vec representations capturing distributional sematics which don't seem to be captured in character level sequences.
- In other words lemma, stem, morphology and inflections convey only a part of word's semantic meaning.
- Purpose is to use C2W2V to create vector embeddings for words not found in word2vec vocabulary.
- Cosine similarity of 0.3680 conveys some meaning and might function as a alternative to using unknown token representation.

4.4 Evaluation

- Recently end-to-end dialogue system have adopted metrics from amchine translation and text summarization. These don't work so well. C.-W. Liu et al. 2016

- C.-W. Liu et al. 2016 considers unsupervised utterance systems where response is generated or selected from a set of possible responses. This is irrelevant since we are building a problem solving dialogue system instead of chattering system.

5. CONCLUSIONS AND FUTURE WORK

BIBLIOGRAPHY

Andor, D. et al. (2016). "Globally Normalized Transition-Based Neural Networks". In: *Acl 2016*, pp. 2442–2452. DOI: 10.18653/v1/P16-1231. arXiv: arXiv:1603.06042v2.

Bahdanau, D., K. Cho, and Y. Bengio (2014). "Neural Machine Translation By Jointly Learning To Align and Translate". In: *Iclr 2015*, pp. 1–15. ISSN: 0147-006X. DOI: 10.1146/annurev.neuro.26.041002.131047. arXiv: 1409.0473. URL: http://arxiv.org/abs/1409.0473v3.

Bhargava, A. et al. (2013). "Easy contextual intent prediction and slot detection". In: *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 8337–8341. ISSN: 15206149. DOI: 10.1109/ICASSP.2013.6639291.

Brill, E. (1992). "A Simple Rule-Based Part of Speech Tagger". In: *Applied natural language*, p. 3. ISSN: 00992399. DOI: 10.3115/1075527.1075553. arXiv: 9406010 [cmp-lg].

Chen, D. and C. D. Manning (2014). "A Fast and Accurate Dependency Parser using Neural Networks". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* i, pp. 740–750. ISSN: 9781937284961. URL: https://cs.stanford.edu/%7B~%7Ddanqi/papers/emnlp2014.pdf.

Cho, K. et al. (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734. ISSN: 09205691. DOI: 10.3115/v1/D14-1179. arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078.

Chung, J., K. Cho, and Y. Bengio (2016). "A Character-level Decoder without Explicit Segmentation for Neural Machine Translation". In: *Acl-2016*, pp. 1693–1703. arXiv: 1603.06147.

Clevert, D.-A., T. Unterthiner, and S. Hochreiter (2015). "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)". In: *Under review of ICLR2016 ELU* 1997, pp. 1–13. arXiv: 1511.07289. URL: http://arxiv.org/pdf/1511.07289.pdf%7B%5C%%7D5Cnhttp://arxiv.org/abs/1511.07289.

Collobert, R. et al. (2011). "Natural Language Processing (Almost) from Scratch". In: *Journal of Machine Learning Research* 12, pp. 2493–2537. ISSN: 0891-2017. DOI: 10.1.1.231.4614. arXiv: 1103.0398.

De Marneffe, M.-C., B. MacCartney, and C. D. Manning (2006). "Generating typed dependency parses from phrase structure parses". In: *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pp. 449–454. DOI: 10.1.1.74.3875. URL: http://nlp.stanford.edu/pubs/LREC06%7B%5C_%7Ddependencies.pdf.

Fernández-González, D. and A. F. T. Martins (2015). “Parsing as Reduction”. In: *In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* 1983, pp. 1523–1533. arXiv: 1503.00030.

Ginter, F. and J. Kanerva (2014). “Fast Training of word2vec Representations Using N-gram Corpora”. In:

Harris, Z. S. (1954). “Distributional Structure”. In: *WORD* 10.2-3, pp. 146–162. DOI: 10.1080/00437956.1954.11659520. URL: http://dx.doi.org/10.1080/00437956.1954.11659520.

Haverinen, K. et al. (2014). “Building the essential resources for Finnish: the Turku Dependency Treebank”. In: *Language Resources and Evaluation* 48.3, pp. 493–531. ISSN: 15728412. DOI: 10.1007/s10579-013-9244-1.

Kestemont, M. et al. (2016). “Lemmatization for variation-rich languages using deep learning”. In: *Digital Scholarship in the Humanities*, fqw034. ISSN: 2055-7671. DOI: 10.1093/llc/fqw034. URL: http://dsh.oxfordjournals.org/lookup/doi/10.1093/llc/fqw034.

Kim, Y. et al. (2016). “Character-Aware Neural Language Models”. In: *Aaai*. arXiv: 1508.06615. URL: http://arxiv.org/abs/1508.06615.

Korenius, T. et al. (2004). “Stemming and lemmatization in the clustering of finnish text documents”. In: *Proceedings of the thirteenth ACM conference on information and knowledge management*, pp. 625–633. DOI: 10.1145/1031171.1031285. URL: http://portal.acm.org/citation.cfm?id=1031171.1031285%7B%5C&%7Dcoll=Portal%7B%5C&%7Ddl=ACM%7B%5C&%7DCFID=88534260%7B%5C&%7DCFTOKEN=49348956.

Li, J. et al. (2015). “A Diversity-Promoting Objective Function for Neural Conversation Models”. In: *Arxiv* Mmi, pp. 110–119. arXiv: 1510.03055. URL: http://arxiv.org/abs/1510.03055.

Liang, C. et al. (2016). “Neural Symbolic Machines: Learning Semantic Parsers on Freebase with Weak Supervision”. In: October. arXiv: 1611.00020. URL: http://arxiv.org/abs/1611.00020.

Ling, W. et al. (2015). “Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation”. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* September, pp. 1520–1530. DOI: 10.18653/v1/D15-1176. arXiv: 1508.02096. URL: http://dx.doi.org/10.18653/v1/d15-1176%7B%5C$%7D%7B%5C%%7D5C%7B%5C$%7Dnfile:///Files/68/6810072d-e133-426e-807f-445df2840420.pdf%7B%5C$%7D%7B%

5C%%7D5C%7B%5C$%7Dnpapers3://publication/doi/10.18653/v1/d15-1176%7B%5C$%7D%7B%5C%%7D5C%7B%5C$%7Dnhttp://arxiv.org/abs/1508.02096.

Liu, B. and I. Lane (2016). "Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling". In: 1, pp. 2–6. DOI: 10.21437/Interspeech.2016-1352. arXiv: 1609.01454. URL: http://arxiv.org/abs/1609.01454.

Liu, C.-W. et al. (2016). "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation". In: *Annual Meeting of the Association for Computational Linguistics (ACL)*, p. 13. arXiv: 1603.08023. URL: http://arxiv.org/abs/1603.08023.

Mikolov, T., G. Corrado, et al. (2013). "Efficient Estimation of Word Representations in Vector Space". In: *Proceedings of the International Conference on Learning Representations (ICLR 2013)*, pp. 1–12. ISSN: 15324435. DOI: 10.1162/153244303322533223. arXiv: arXiv:1301.3781v3. URL: http://arxiv.org/pdf/1301.3781v3.pdf.

Mikolov, T., W.-t. Yih, and G. Zweig (2013). "Linguistic regularities in continuous space word representations". In: *Proceedings of NAACL-HLT* June, pp. 746–751. URL: http://scholar.google.com/scholar?hl=en%7B%5C&%7DbtnG=Search%7B%5C&%7Dq=intitle:Linguistic+Regularities+in+Continuous+Space+Word+Representations%7B%5C#%7D0%7B%5C%%7D5Cnhttps://www.aclweb.org/anthology/N/N13/N13-1090.pdf.

Nivre, J. (2004). "Incrementality in deterministic dependency parsing". In: *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pp. 50–57. DOI: 10.3115/1613148.1613156. URL: http://dl.acm.org/citation.cfm?id=1613156.

Nivre, J. et al. (2016). "Universal Dependencies v1: A Multilingual Treebank Collection". In: *Proceedings of the 10th International Conference on Language Resources and Evaluation (LREC 2016)*, pp. 1659–1666.

Pennington, J., R. Socher, and C. D. Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1532–1543. ISSN: 10495258. DOI: 10.3115/v1/D14-1162. arXiv: 1504.06654.

Petrov, S., D. Das, and R. Mcdonald (2012). "A Universal Part-of-Speech Tagset". In: arXiv: 1104.2086.

Pyysalo, S. et al. (2015). "Universal Dependencies for Finnish". In: *Nordic Conference of Computational Linguistics NODALIDA 2015* Nodalida, p. 163.

Sutskever, I., O. Vinyals, and Q. V. Le (2014). "Sequence to sequence learning with neural networks". In: *Nips*, pp. 1–9. ISSN: 09205691. DOI: 10.1007/s10107-014-

0839-0. arXiv: 1409.3215. URL: http://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural.

Takala, P. (2016). "Word Embeddings for Morphologically Rich Languages". In: April, pp. 27–29.

Weiss, D. et al. (2015). "Structured Training for Neural Network Transition-Based Parsing". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* 2012, pp. 323–333. DOI: 10.3115/v1/P15-1032. arXiv: 1506.06158. URL: http://www.aclweb.org/anthology/P15-1032.

Xu, P. and R. Sarikaya (2013). "Exploiting shared information for multi-intent natural language sentence classification". In: *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH* 1, pp. 3785–3789. ISSN: 19909772.

Zhang, X. and Y. LeCun (2015). "Text Understanding from Scratch". In: *APL Materials* 3.1, p. 011102. ISSN: 2166-532X. DOI: 10.1063/1.4906785. arXiv: 1502.01710. URL: http://arxiv.org/abs/1502.01710.

APPENDIX A. SOMETHING EXTRA

Appendices are purely optional. All appendices must be referred to in the body text

APPENDIX B. SOMETHING COMPLETELY DIFFERENT

You can append to your thesis, for example, lengthy mathematical derivations, an important algorithm in a programming language, input and output listings, an extract of a standard relating to your thesis, a user manual, empirical knowledge produced while preparing the thesis, the results of a survey, lists, pictures, drawings, maps, complex charts (conceptual schema, circuit diagrams, structure charts) and so on.