Empirical Analysis of LSTM, CRF and Attention on the CONDA dataset.

Johnathan Mc Donogh

[1] University of Sydney, NSW 2006, Australia
jomc7031@uni.sydney.edu.au

1 Data preprocessing

No content.

2 Input Embedding

Input word vectors were generated in three dimensional aspects: Syntactical, Semantic and a specialized Domain feature embedding. Embedding creation is a domain sensitive technique and therefore the approach for each aspect considered the general semantic and syntactic language usage within the domain (Shi et al. 2018, p2). For the CONDA (Weld, et. al. 2021) dataset, language usage can be described as concise and informal. Grammatical rules and semantic accuracy are not strictly followed in the domain of in-game chats. Instead, brevity supported by domain specific acronyms, contractions and slang are preferred as they support conveyance of meaning in brief sequences. Secondly, in-game discourse does not involve the complex exchange of ideas or information which generally requires complex sentence structures, speech elements and syntactic construction. Instead, the CONDA discourse could be described as basic reactionary, action based, commentary to domain specific in-game events.

For **syntactical feature** embedding two methods were considered, Part of Speech (POS) tags and Syntactic Dependency Parsing. The key point of difference between POS and dependency parsing is that Dependency Parsing establishes a syntactic relationship between parts of speech while POS ends at their identification (Jurafsky & Martin 2021, p1). Colloquialism, errors in text and speech, slang and domain specific language present challenges to syntactical feature extraction methods given that informal language is often replete with grammatical and spelling errors and does not conform linguistic rules which these models rely upon (Derczynski et. al. 2013, p1-2). Moreover, domain specific Dota terminology in not within the scope of these models and as such identification of dependency paths and speech elements may not be possible (Derczynski et. al. 2013, p1-2). However, in the specific context, pronoun entity identification is important to the model as the outcome requires identification of pronoun elements. Input features that support better differentiation pronoun entities from other discourse element may improve model performance.

Overall syntactical feature extraction method appears to be a challenge for the current domain owing to the informal nature of in-game speech and its concentration of slang, colloquialisms and domain specific terms. As Dependency Parsing places a greater reliance on conformance to grammatical rules than POS in order to establish syntactic relationships, it was considered relatively more unsuitable to the current domain. Notwithstanding the challenges of POS for this dataset, if we accept that in-game chat could generally be described as reactionary, action based, commentary, then it is not evident that creation of complex syntactic relationships between words will be beneficial in an informal language context. Instead, the basic extraction of verbs, adverbs and most importantly, *pronouns,* where possible may provide the greatest benefit particularly when merged with semantic, domain specific, features. Ideally, a domain specific model that incorporates nouns and elements specific to Dota would be most suitable but is beyond the scope of this paper. The POS was implemented using the Penn Treebank POS tagger (Marcus et. al. 1992, p1).

For **semantic feature** embedding the following approaches were considered: Word2Vec, FastText, GLoVE and ELMO. ELMO (and BERT) differ from the other embeddings in that ELMo generates context dependent vector expression for a word whereas Word2Vec, FastText and GLoVE are context independent and generate the same vector expression regardless of context (Peters et. al. 2018, p1-2). Empirical studies show that ELMo models complex syntactic and semantic characteristic of word usage and performs well in natural language environment where the proliferation of homonyms, permutations of speech construction and subtle syntactic variations results in complex word usage (Peters et. al. 2018, p1-2). Conversely, the language environment of in-game chat is unlikely to be described as semantically and syntactically complex where complex word expression may evolve from its usage in different variations. Given the informal and uncomplex syntactic construction of in-game chat the incremental benefit of utilizing complex context word expressions may be limited. For these reasons, ELMo was ruled out.

Next the context independent models were considered and ultimately FastText was selected. This choice was driven by FastText robust capability to address misspelling, a capability which the other two lack. Misspelling is an important factor in informal language setting and one of the key advantages FastText offers over other non-contextual embeddings (Athiwaratkun et. al. 2018, p1-2). Compared to Word2Vec and GLoVE, FastText also has demonstrable benefits in generating better expression for rare words in a corpus (Athiwaratkun et. al. 2018, p1-2). This is important owing to the dominance of 'O' features and the requirement for the model to be sensitive to potentially subtle and delitescent semantic difference in other less common features such as 'T' and 'D'. As a result, FastText was deemed more suitable to the natural of the domain dataset given its proliferation of misspellings, informal language and the need to be sensitive to rare words within the corpus. The FastText model was generated using the merged training and test corpus.

Finally, a **domain feature embedding** was considered. Once again, the semantic and syntactic nature of language usage within the domain was considered with particular attention to its basic characteristic of brevity, slang, colloquialism and domain specific language usage. Each characteristic was addressed in the following way to build a domain specific embedding:

1) The semantic basis of the feature embedding is based on a FastText vector generated from the supplied CONDA corpus (as outlined in semantic aspect above) which was extensively supplemented with a large corpus derived from obtaining Dota related web pages comprising the subjects listed below. The objective of obtaining additional information was to create a larger representative corpus of the domain with particular emphasis on obtaining the usage of rare and Dota specific terms, acronyms and colloquialism in order to build better semantical expressions and semantic relationships relevant to the Dota domain. The anticipated outcome is that the downstream model will be better at separating the Dota elements (heroes and terms) and their associative acronyms from general discourse. The extended Dota corpus comprises discourse and data over the following topics:
    i) Basic descriptions, strategy and general gameplay guides for all Dota heroes
    ii) Glossaries of in-game Dota specific terms and common Dota acronyms.
    iii) Discussion pages and blogs of Dota gameplay and beginner user guides.
2) The glossaries of Dota heroes, common acronyms and terms derived in the previous section were further used to generate domain specific one-hot-encoded vectors aimed at creating features that further separate Dota characters and gameplay terms. These vectors were concatenated to the feature vector derived in section one above and comprise:
    i) Is_dota_hero: identifies whether word is a known Dota hero
    ii) Is_dota_acronym: identifies whether word is a common gameplay term or object
3) Next an additional one-hot-encoded feature vector was added to identify profane or offensive terms in the corpus and thereby enable better sensitivity to toxicity ('T') elements. These features were identified using a profanity word list built specifically for gaming contexts and its associative slang (e.g. noob) from the Luis von Ahn (no date, p1) research group .
    i) Is_profanity: identifies whether subject word is a known offensive or profane term.
4) Finally, the feature vector was supplemented with pronoun identification using the POS tagger from the syntactic aspect section outlined above. This was added with the objective to enable better sensitivity to pronoun entity recognition.
    i) Is_pronoun: identifies whether subject word is pronoun part of speech using Upenn Treabank POS tagger (Marcus et. al. 1992, p1).

In summary and for the purpose of input embedding into the models described in the following section three input feature vectors were constructed: (1) FastText only trained from supplied CONDA corpus, (2) FastText trained from supplied CONDA dataset concatenated with POS tags and (3) FastText trained from supplied corpus extended with Dota related website data and concatenated with domain specific features.

3 Slot Filling/Tagging Models

Prediction of the CONDA tags was implemented with seven experimental variations of a LTSM/GRU recurrent neural network separated in approach by three architectural modifications comprising (a) Bi-LSTM with various stacks (layers), (b) GRU with various attention calculations and (c) GRU with Attention and Conditional Random Fields (CRF). Parameter selection, training epochs, setup environment, input embedding dimensions, learning rates and optimizers are addressed in the (next) evaluation section. The purpose of this section is to deductively explain the modifications above and the justification for each.

3.1 Bi-LTSM Layer Stacking

The baseline LSTM model design comprises a single hidden LSTM layer followed by a feedforward output layer. Stacked LSTM extends this foundation, creating a deeper network of multiple hidden layers. Deep networks and deep learning are attributed to incremental success in various domains with the additional layers understood to recombine learned representations from prior layers creating new and more powerful representations at incrementally higher levels of abstraction to the benefit of prediction power (Hermans & Chrauwen 2013, p1). Specific to the LSTM architecture that operates on sequence data, this means adding layers with theoretically result in higher levels of abstraction over time and potentially allow hidden states at each level to operate over different timescales (Pascanu et. al. 2014, p5).

Increased LSTM layers does not guarantee performance improvements: Recurrent networks are inherently deep owing to their propagation of previous states and therefore incremental vertical layers *may* be beneficial (Graves et. al. 2013, p1). Empirical studies show that deep recurrent network perform better than shallower ones on some tasks (Hermans & Chrauwen 2013, p1). Following the literature, it appears the important aspect to consider is the implementation domain and its complexity: roughly speaking, the complexity and depth of the model should be commensurate with the complexity data or domain application and its intended outcome.

Considering both the literature and complexity of task at hand, it was considered that, compared to most natural language scenarios and settings, the Dota dataset and target outcome may not be complex to the degree that it would warrant very deep neural models. The addition of LSTM layer may also introduce vanishing gradient issues leading to performance degradation (Hochreiter 1998, p107). As such the approach taken to stacked layer modifications was to begin at 1 LSTM layer and stack only 1 additional LSTM layer of the same hidden density. The outcome of this approach is documented in the evaluation section.

3.2 GRU with Attention

The encoder-decoder architecture in this scenario comprises one GRU network that learns or encodes input game chat sequences as a fixed length internal representation and a second network that decodes or converts the encoded input into the tag sequence. A potential issue with the standalone encoder-decoder approach is that the network must compress all necessary information from source chat sequences of varying lengths into a fixed-length (internal) vector (Bahdanau et. al. 2016, p1). This may impact the model performance for long chat sequences or chat sequences in the test dataset that are longer than what is found the supplied CONDA training dataset and exacerbate vanishing gradient problems.

The objective of attention is to free the encoder-decoder architecture from fixed-length internal vector representations (Bahdanau et. al. 2016, p1). Attention, searches for a set of position in the source input sequences where the most relevant information is concentrated (Bahdanau et. al. 2016, p1). Using the context vectors associated with this position of the input sequence, the model predicts the target and as a result no longer needs to squeeze all information of source sentence into a fixed vector (Bahdanau et. al. 2016, p1). In the application of sequence prediction, it has been found to generate superior results given the model is freed from encoding the entire source sentence and focuses only on information relevant to generation of next tag (Bahdanau et. al. 2016, p1).

The attention score mechanisms formed the basis of this experiment. A single encoder-attention-decoder module was used to generate predictions using three different attention score mechanism: (a) Dot product of encoder and decoder hidden states. This method was chosen following the demonstrable performance gains by Luong et. al. (2015, p7) in their paper Effective Approaches to Attention-based Neural Machine Translation. (b) Scaled dot product (result scaled to squared dimension of hidden states) was selected following research and favorable results by Vaswani et. al. (2017, p4) and, the final scoring mechanism, (c) content-based cosine scoring (Graves et. al. 2014, p8). The cosine method was chosen following the findings of Graves (2014, p8) which attenuates the precision of the focused vectors by similarity measure. The decision to position attention between the encoder and decoder followed the placement structure described by Chen, Bolton & Manning (2016, p2361). This structure is broadly analogous to the *AttentiveReader* (Hermann et. al. 2015, p1684) and was reference architecture for Chen et. al.

3.3 GRU with Attention and CRF

In an extension to the prior section the model with the single encoder-decoder pair and best attention score calculation method is attached with Conditional Random Fields. CRF has demonstrable application to relational learning task where the contextual information or state of neighbors affect the current prediction and has good application to speech tagging (Song et. al 2019, p1). This is reflective in model performance addressed in the following section.

4 Evaluation

4.1 Evaluation setup

This paper evaluates input embeddings and seven experimental neural network variations in a *four-part* ablative study: (1) Input embedding performance of syntactic, semantic and domain specific inputs using a consistent baseline LSTM CRF model. (2) Attention score performance across the three score methods described in section 4.2 given the placement of attention as described in the same section. (3) Performance of the stacked Bi-LSTM model across variations in vertical LSTM stacks and finally, (4) Performance of the Attention model described in section 4.2 before and after the application of CRF.

The ablative study setup including environment, parameters, training epochs, hidden layers, embedding dimensions and optimizer, where applicable, are summarized in the experimental setup table below.

Part 1
Input Embedding Comparison

Reference	Variation Name	Creation Environment	Primary API	Parameters	Training Epochs	Measure
Section 1	FastText + POS (one-hot-econding)	Python - Google Colab	FastText, NLTK	FastText embedding dimension = 100 POS dimension = 41 Total dimensions = 141	N/A	Validation F1 Score
Section 1	FastText Plain	Python - Google Colab	FastText	FastText embedding dimension = 100	N/A	Validation F1 Score
Section 1	Domain Specific (see section description)	Python - Google Colab	FastText, NLTK, custom APIs	FastText embedding dimension = 100 custom feature dimensions = 4 Total dimensions =104	N/A	Validation F1 Score

Part 2
Attention Score Strategies

Reference	Variation Name	Creation Environment	Primary API	Parameters	Training Epochs	Measure
Section 3.2	GRU Attention: **Dot Product score**	Python - Google Colab	Pytorch	hidden layers = 50 Input embedding = Domain Specific Optimizer = SGD GRU stacks = 1 Attention position = between encoder and decoder Learning rate: 0.1	2	Validation F1 Score
Section 3.2	GRU Attention: **Scaled Dot Product score**	Python - Google Colab	Pytorch		2	Validation F1 Score
Section 3.2	GRU Attention: **Cosine score**	Python - Google Colab	Pytorch		2	Validation F1 Score

Part 3
Stacked LSTM Layers

Reference	Variation Name	Creation Environment	Primary API	Parameters	Training Epochs	Measure
Section 3.1	Bi-LSTM: **1 LSTM** Layer	Python - Google Colab	Pytorch	hidden layers = 50 Input embedding = Domain Specific Optimizer = SGD Learning rate: 0.1	2	Validation F1 Score
Section 3.1	Bi-LSTM: **2 LSTM** Layers	Python - Google Colab	Pytorch		2	Validation F1 Score

Part 4
LSTM CRF

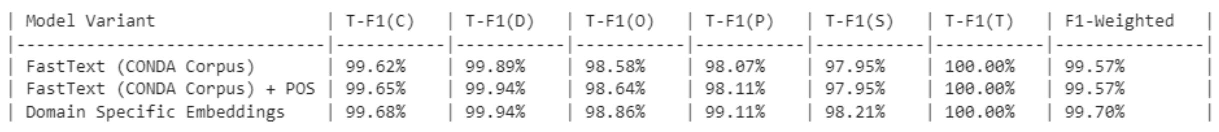
Reference	Variation Name	Creation Environment	Primary API	Parameters	Training Epochs	Measure
Section 3.3	GRU Attention: **non-CRF** (from section 3.2)	Python - Google Colab	Pytorch	hidden layers = 50 Imput embedding = Domain Specific Optimizer = SGD, Learning rate: 0.1 LSTM stacks = 1 Attention position = between encoder and decoder	2	Validation F1 Score
Section 3.3	GRU Attention: **CRF**	Python - Google Colab	Pytorch		2	Validation F1 Score

4.2 Evaluation result

4.2.1: Ablation study part 1: input embedding models

The summative outcome of the input embedding comparison using the baseline model (Bi-LSTM CRF) is summarized in the table below. Although not enough experimental repetitions were performed to assess the statistical significance of the variations, F1 performance of the Customized Domain embedding described in the opening section of this report stands out as the strongest performer and was the best model across all other studies in this paper. The trend from the table suggests that building a more expressive word embedding using syntactical (POS) and customized features may contribute to incremental model performance gains due to improved word expression. Following the advice of Lai et. al. (2015, p2) the focus of the Domain Specific Embeddings was to build domain relevant features (and extended corpus) rather than focus on corpus size and vector expression dimensions.

Table 1 Metrics: Emmbedding performance on baseline Bi-LSTM CRF Model



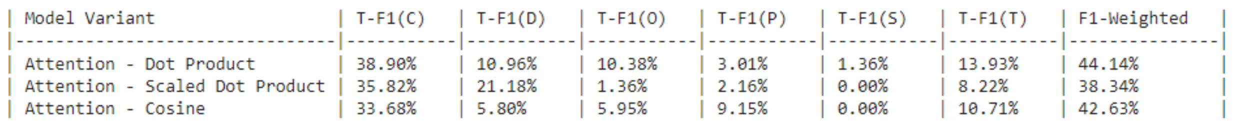
```
| Model Variant                  | T-F1(C)   | T-F1(D)   | T-F1(O)   | T-F1(P)   | T-F1(S)   | T-F1(T)   | F1-Weighted   |
|--------------------------------|-----------|-----------|-----------|-----------|-----------|-----------|---------------|
| FastText (CONDA Corpus)        | 99.62%    | 99.89%    | 98.58%    | 98.07%    | 97.95%    | 100.00%   | 99.57%        |
| FastText (CONDA Corpus) + POS  | 99.65%    | 99.94%    | 98.64%    | 98.11%    | 97.95%    | 100.00%   | 99.57%        |
| Domain Specific Embeddings     | 99.68%    | 99.94%    | 98.86%    | 99.11%    | 98.21%    | 100.00%   | 99.70%        |
```

Interestly, it appears that incorporation of the profane data set resulted in material improvement to the model's ability to differentiate offensive terms that lead to toxic chat interactions. Although the Domain Specific Embedding Model was better at identifying pronouns than the base FastText Model (trained on corpus only), it failed to outperform the other models on Dota term identification. This suggests that the feature embeddings from Dota 2 glossaries are not reflective of in game chats or perhaps not extensive enough. In summary, results appear to support the general heuristic that the embeddings used to train the model should be analogous with the context of the problem (Lai et. al. 2015, p2). In this case creating a customized embeddings with features targeted at supporting identification of the target outcome demonstrated material improvement on a basic FastText trained vector representation.

4.2.2: Ablation study part 2: Attention score strategies

Three attention scoring mechanisms were used: Dot Product, Squared Dot Product and Cosine. The justification for this selection is outlined in section 3.2. The table demonstrates that the inclusion of the attention architecture results in a degradation of performance when compared to the baseline model of stacked Sequence 2 Sequence models. This outcome was unexpected as attention was thought to improve entity tagging tasks for sequence inputs; however, this may be attributable to poor implementation and optimization.

Table 3 Metrics: GRU Attention Based Scores



```
| Model Variant                  | T-F1(C)   | T-F1(D)   | T-F1(O)   | T-F1(P)   | T-F1(S)   | T-F1(T)   | F1-Weighted   |
|--------------------------------|-----------|-----------|-----------|-----------|-----------|-----------|---------------|
| Attention - Dot Product        | 38.90%    | 10.96%    | 10.38%    | 3.01%     | 1.36%     | 13.93%    | 44.14%        |
| Attention - Scaled Dot Product | 35.82%    | 21.18%    | 1.36%     | 2.16%     | 0.00%     | 8.22%     | 38.34%        |
| Attention - Cosine             | 33.68%    | 5.80%     | 5.95%     | 9.15%     | 0.00%     | 10.71%    | 42.63%        |
```

With respect to scoring mechanisms, the table demonstrates a trend of performance that does not align with the literature: Dot Product attention scoring was the strongest scoring mechanism followed by Cosine and finally Scaled Dot Product. Following the findings of Luong (2015, p7) and Vaswani (2017, p4) the expected performance outcome was for scaled and non-scaled dot product to outperform the older content-based cosine scoring mechanism. Furthermore, the scaled dot product was expected to be the strongest performer (Vaswani, 2017, p4). Further investigation would be required to determine the reasons for the unexpected performance outcomes.

4.2.3: Ablation study part 3: Stacked LSTM layers

In summary, the addition of a second LSTM layer resulted in a minor degradation of performance with no improvement in tag sensitive across all tag types. This trend seems to support the notion that the introduction of additional layers may exacerbate vanishing or exploding gradient issues commonly encountered in deeper network and which lead to performance degradation (Hochreiter 1998, p107). As a result, this may impact the ability for the network to propagate useful information contributing to the performance decline (Hochreiter 1998, p107).

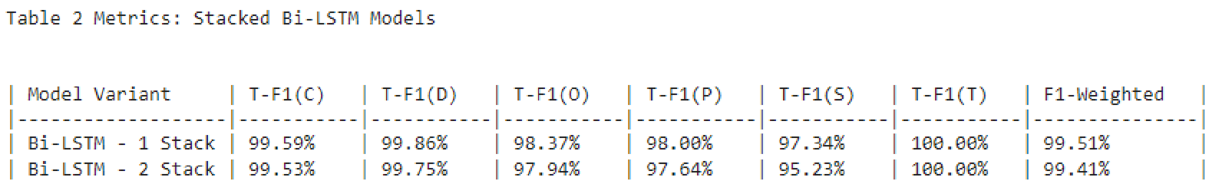


Table 2 Metrics: Stacked Bi-LSTM Models

Model Variant	T-F1(C)	T-F1(D)	T-F1(O)	T-F1(P)	T-F1(S)	T-F1(T)	F1-Weighted
Bi-LSTM - 1 Stack	99.59%	99.86%	98.37%	98.00%	97.34%	100.00%	99.51%
Bi-LSTM - 2 Stack	99.53%	99.75%	97.94%	97.64%	95.23%	100.00%	99.41%

In a separate but similar vein, the performance decline may also suggest that higher order representation of the data across separate timescales - which stacked layers confer (Pascanu et. al. 2014, p5) - may not be suited to the data and task at hand.

4.2.4: Ablation study part 4: CRF/non-CRF

CRF has demonstrable application to relational learning task where the contextual information or state of neighbors affect the current prediction and has application in speech tagging such as this (Song et. al 2019, p1). Indeed, the excellent performance of the baseline CRF model shows that CRF is advantageous over the standard Bi-LSTM architecture. Unfortunately, the technical implementation of Attention and CFR was not achieved, however, based on the above-mentioned studies and the empirical results CRF is expected to result in incremental performance gains.

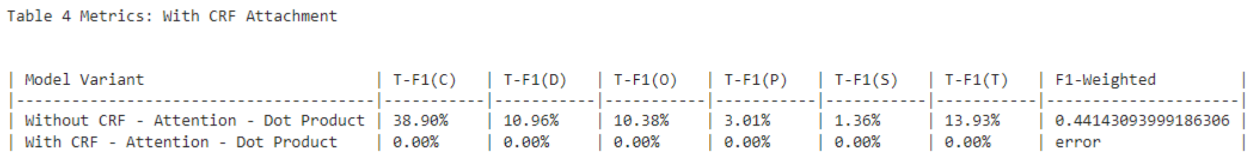


Table 4 Metrics: With CRF Attachment

Model Variant	T-F1(C)	T-F1(D)	T-F1(O)	T-F1(P)	T-F1(S)	T-F1(T)	F1-Weighted
Without CRF - Attention - Dot Product	38.90%	10.96%	10.38%	3.01%	1.36%	13.93%	0.44143093999186306
With CRF - Attention - Dot Product	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	error

References

1. S. Khalid, T. Khalil and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," 2014 Science and Information Conference, 2014, pp. 372-378, doi: 10.1109/SAI.2014.6918213. https://www.researchgate.net/publication/287743399_A_survey_of_feature_selection_and_feature_extraction_techniques_in_machine_learning

2. Bei Shi, Zihao Fu, Lidong Bing, Wai Lam. "Learning Domain-Sensitive and Sentiment-Aware Word Embeddings". 2018. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 2494–2504. https://aclanthology.org/P18-1232.pdf

3. Daniel Jurafsky & James H. Martin. Speech and Language Processing. 2021. Stanford University. https://web.stanford.edu/~jurafsky/slp3/14.pdf

4. Leon Derczynski, Alan Ritter, Sam Clark, Kalina Bontcheva."Twitter Part-of-Speech Tagging for All: Overcoming Sparse and Noisy Data". 2013. Proceedings of Recent Advances in Natural Language Processing, pages 198–206. https://aclanthology.org/R13-1026.pdf

5. Matthew E. Peters , Mark Neumann , Mohit Iyyer , Matt, Christopher Clark , Kenton Lee , Luke Zettlemoyer. Gardner Deep contextualized word representations. 2018. Allen Institute for Artificial Intelligence. Paul G. Allen School of Computer Science & Engineering, University of Washington. https://arxiv.org/abs/1802.05365

6. Ben Athiwaratkun, Gordon Wilson.Anima Anandkumar. Probabilistic FastText for Multi-Sense Word Embeddings 2018. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 1–11. https://aclanthology.org/P18-1001.pdf

7. Mitchell P. Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz. 1992. Building a large annotated corpus of English: the Penn Treebank. https://catalog.ldc.upenn.edu/docs/LDC95T7/cl93.html

8. Luis von Ahn. Offensive/Profane Word List. University of Carnegie Mellon. https://www.cs.cmu.edu/~biglou/resources/

9. Henry Weld, Guanghao Huang, Jean Lee, Tongshu Zhang, Kunze Wang, Xinghong Guo, Siqu Long, Josiah Poon, Soyeon Caren Han . 2021. CONDA: a CONtextual Dual-Annotated dataset for in-game toxicity understanding and detection. https://arxiv.org/abs/2106.06213

10. Charles Sutton and Andrew McCallum. 2011. An Introduction to Conditional Random Fields. https://homepages.inf.ed.ac.uk/csutton/publications/crftut-fnt.pdf

11. Song, S., Zhang, N. & Huang, H. 2019. Named entity recognition based on conditional random fields. Cluster Comput 22, 5195–5206 (2019). https://doi.org/10.1007/s10586-017-1146-3

12. Michiel Hermans, Benjamin Schrauwen. 2013. Training and Analysing Deep Recurrent Neural Networks. https://papers.nips.cc/paper/2013/hash/1ff8a7b5dc7a7d1f0ed65aaa29c04b1e-Abstract.html

13. Alex Graves, Abdel-rahman Mohamed, Geoffrey Hinton. 2013. Speech Recognition with Deep Recurrent Neural Networks. https://arxiv.org/abs/1303.5778

14. Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. https://arxiv.org/abs/1409.0473

15. Alex Graves, Greg Wayne, Ivo Danihelka. 2014. Neural Turing Machines. Google DeepMind. https://arxiv.org/pdf/1410.5401.pdf

16. Minh-Thang Luong, Hieu Pham, Christopher D. Manning. "Effective Approaches to Attention-based Neural Machine Translation". 2015. https://arxiv.org/pdf/1508.04025.pdf

17. Siwei Lai, Kang Liu, Liheng Xu, Jun Zhao. 2015. How to Generate a Good Word Embedding? https://arxiv.org/abs/1507.05523

18. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser. 2017. Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017). https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

19. Hochreiter, Sepp. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 6. 107-116. 10.1142/S0218488598000094.

20. Danqi Chen and Jason Bolton and Christopher D. Manning. 2016. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 2358–2367. https://aclanthology.org/P16-1223.pdf.

21. Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Yoshua Bengio. 2014. How to Construct Deep Recurrent Neural Networks. https://arxiv.org/abs/1312.6026

22. Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In Advances in Neural Information Processing Systems

(NIPS), pages 1684–1692. https://papers.nips.cc/paper/2015/hash/afdec7005cc9f14302cd0474fd0f3c96-Abstract.html