Neural Attention and Morphological Word Embedding for Contract Element Extraction

Department of Data Science and Knowledge Engineering
Maastricht University

José Luis Velásquez Sosa
Tutor: Gerasimos Spanakis

June 28, 2018

Abstract

Contract element extraction tries to identify the most important facts and entities of a contract in order to keep an accurate track of a legal corpus. The introduction of neural attention mechanism and morphological word embedding are added to the previously proposed element extractors,the Bidirectional Long-Short Term Memory network with a Conditional Random Field (BiLSTM-CRF) and the stacked BiLSTM-LSTM with a logistic regression as a output(BiLSTM-LSTM-LR), to explore their effect on the extracting task. No significant effect was found, on the introduction of attention mechanism. However, statistically significant results with a p-value of 0.006 suggests that the introduction of morphological embedding helps the BiLSTM-CRF and BiLSTM-LSTM-LR architectures perform better.

1 Introduction

As organizations try to automate their internal and external processes as much as possible, the need to keep track of legal obligations, rights and limitations has become an important task. For instance, to keep track of the information of company-provided car leasing contracts, it would be wise to track important elements in the contract such as dates of payment, product value, termination conditions, etc. Keeping an accurate track of legally binding contracts can help companies better prepare for audits and prevent ambiguous contract breaches among many other things. However, the process of keeping track of an entire corpus of legal documents can be a time-costly task. Consequently, different automated contract element extractors have been proposed to extract individual elements from a contract. In past works on the contract element extraction task, the BiLSTM architecture, which is later explained, has shown to perform better than non-deep learning extractors and almost matched by the performance of the BiLSTM-LSTM-LR architecture.

In this paper neural attention mechanisms and morphological embedding are proposed to enhance the BiLSTM-CRF architecture to explore the following research questions:

1. How does a neural attention mechanism affect the BiLSTM-CRF architecture on the contract element extraction task?

2. How does morphologically-aware word embedding affect the performance of the BiLSTM-LSTM-LR and BiLSTM-CRF architecture on the contract element extraction?

3. Do different extractors perform better on different super-entities?

2 Literature Review

The contract element extraction task can be modelled as a binary sequence tagging problem where words are tagged either as an extracted element or not. Therefore, the proposed extractors use techniques known for their good performance in sequence tagging. Two sets of extractors have been proposed and tested so far on the same dataset. The first set used hand crafted rules in conjunction with Logistic Regression (LR) and Support Vector Machines (SVM) [2]. From this set of extractors, the ones with the best performance were a combination of SVM and manually written post-processing rules. The second set of proposed extractors consisted of Deep Learning architectures including stacked Bidirectional Long-Short Term Memory (BiLSTM) networks, Long-Short Term Memory (LSTM) networks, and Conditional Random Fields (CRF)[15]. It was shown that of all the previously proposed extractors the BiLSTM-LSTM-LR and BiLSTM-CRF performed best.

The BiLSTM-CRF and BiLSTM-LSTM-LR extractors are composed of several deep learning components, including Long-Short Term Memory, Bidirectional networks and Conditional Random Fields. In addition to the components which compose the previous extractors, more deep learning techniques have shown to aid sequence tagging tasks. The first addition is neural attention. Neural attention mechanisms have shown to produce fruitful results in sequence classification [13] and in sequence tagging [9] as they assist Recurrent Neural Networks (RNN) to better detect and enhance features of the data that is being handled. The second addition is Convolutional Neural Networks (CNNs), which have being shown to be effective when performing feature detection.

Long-Short Term Memory (LSTM)

The LSTM architecture is a special type of RNN. Meaning that its past state is also taken into consideration when fitting and predicting data. The LSTM architecture differs from a normal RNN because it uses a gated cell which can store, read or write information[7]. The advantages of this architecture are that by using the gated cell the network learns to identify inputs that will affect the memory state. This way memory can be more reliable and information can last longer in the cell. However, because of the mechanism used by the LSTM, training can take longer than with conventional RNNs. Furthermore, LSTM networks are prone to over-fitting. To avoid this, a hyperparameter of dropout rate is used [14]. The dropout rate is a probability at which connections are deactivated during training, helping avoid overfitting.

Bidirectional-LSTM (BiLSTM)

The BiLSTM architecture consists of two stacked LSTM layers, with the first reading the sequence from beginning to end and the second from end to beginning. This allows the network to not only take into account the previous information in the sequence, but also future information. Using this bidirectional approach has shown to reduce training time as more semantic information is extracted, and a richer representation of the sequence as a whole is analyzed. As the BiLSTM architecture consists of two LSTM layers, the use of a dropout rate is also commonly used to avoid overfitting.

Conditional Random Field (CRF)

CRF is a graphical statistical method that takes advantage of the Markov Property $p(X_n = x_n|X_{n-1} = x_{n-1}, ..., X_0 = x_0) = p(X_n = x_n|X_{n-1} = x_{n-1})$ where X_i is the ith random variable and x_i is the ith observation of a sequence. The CRF model tries to predict the label Y_i given the observation X_i and prediction Y_{i-1}. The probability distribution $p(Y_i|X_i, Y_{i-1})$ is learned by optimizing the parameters θ via maximum likelihood. This algorithm has been shown to perform richly in different sequence tagging tasks like NER and POS-tagging. Furthermore, the CRF algorithm can be modeled as a type of RNN [16] which can be added as the last layer of a deep learning architecture, taking the last prediction and architecture output as its input.

Neural Attention

Attention mechanisms are augmentations on a RNN which allow the network to also learn which elements in a sequence are more relevant to the task. The relevance is determined by giving some type of weight to each feature vector[12]. In Figure 2 it is shown how the attention mechanism works. The attention mechanism takes the input i and determines an attention weight α_i. Afterwards, the BiLSTM output $V_i b$ gets scaled by the α_i. Therefore, the input for the CRF layer is $\alpha_i V_i$. Several mechanisms have being proposed to calculate α_i [13]. In this paper the mechanism proposed is: $\alpha_i = \overline{\sigma(O_i)}$ were σ is the sigmoid function and O_i is the output of a fully connected layer which takes as input the input of the BiLSTM module.

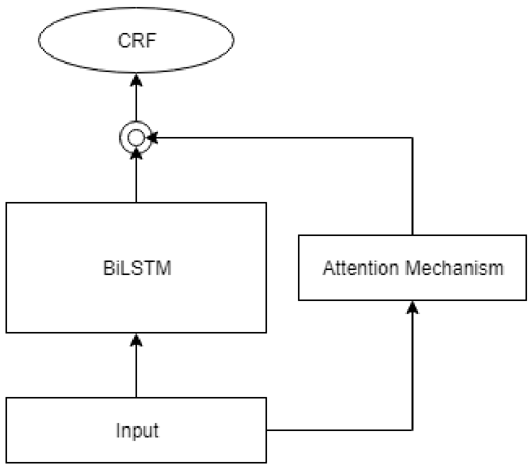

Figure 1: Attention Mechanism around BiLSTM

2.1 Convolutional Neural Network (CNN)

CNNs are a type of deep learning architecture which trains convolutional filters. These filters are useful for feature detection. Therefore, applying convolutional filters to text data can extract features of a word by filtering through the words that were before and after it. The resulting on a richer feature vector that can be used as the new embedding of the word itself. This type of embedding can be applied on a character and word level. Both of these embeddings have shown to correlate new words with human judgment, which shows this is an effective way of embedding words of a natural language[1].

3 Dataset

The dataset used to evaluate this task consists of 2461 annotated and encrypted contracts with 11 different types of element extraction [1]. . However, this paper will focus on extractors of five of the elements:

- Parties (CNP)
- Start Date (STD)
- Effective Date (EFD)
- Termination Date (TED)
- Governing Law (GOV)

The encryption of the contracts works by giving each word a token. The token is a reference to a dictionary with a 200-dimensional embedded vectors obtained from a pre-trained word2vec[11] model which used approximately 750,000 contracts in its training. The token also contains a part-of-speech (POS) tag. Each POS tag has a 25-dimensional pre-trained embedded vector in the dictionary. Therefore, token $t \in R^{225}$ and document $d \in R^{n_d \times 225}$, where n_d is the number of words in the document. However, in this dataset the shape of the word itself is unknown. The shape of the word means wether there are capital letters, numbers, dashes or any other type of character-level information that could be useful for the element extraction task. The lack of word-shape information could make the training tougher as character-level information has being shown to be important for similar tasks like Named Entity Recognition(NER) and POS-tagging [3] [4].

4 Proposed Extractors

In order to perform the task, different extractors are proposed. Each extractor is in itself

[1]The dataset can used can be found in: http://nlp.cs.aueb.gr/software_and_datasets/CONTRACTS_ICAIL2017

a stacked deep learning architecture using a combination of the simpler architectures presented in Section 2. Each extractor extracts one contract element. Therefore new extractors will be made for each contract element that is to be extracted.

4.1 BiLSTM-LSTM-LR

This extractor consists of a BiLSTM layer with an LSTM stacked on top of it. On top of the LSTM there is a fully connected layer with one output. This output computes a probability by using Logistic Regression(LR) through the function $\sigma(x) = \frac{1}{1+e^{x}}$. If $output(extractor) > \pi$, where π is a probability threshold, then the extractor will classify the word as a positive. The probability threshold is individually selected for every extractor, in order to increase performance. An unfolded version of the model is shown in Figure 4.1.

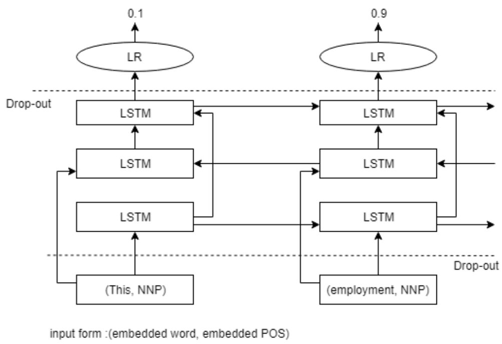

Figure 2: BiLSTM-LSTM-LR

4.2 BiLSTM-CRF

This extractor consists of a BiLSTM network with a CRF, modeled on an RNN, on top of it. The output and evaluation are presented in the same format as the one from the extractor above. However, these probabilities are computed using the CRF layer. The biggest difference in this architecture is that all modules in the network are recurrent, which may prove to be an advantage over the BiLSTM-LSTM-LR.

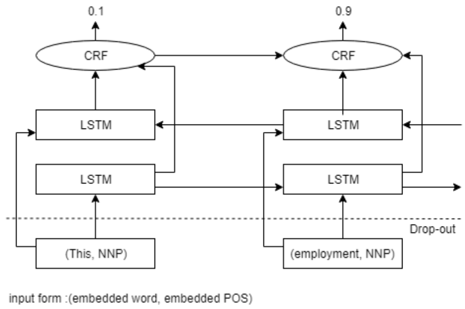

Figure 3: BiLSTM-CRF

The BiLSTM-CRF model has been shown to perform better over all classes in the contract element extraction dataset [15].

4.3 Att-BiLSTM-CRF

The Att-BiLSTM-CRF extractor is the same as the BiLSTM-CRF extractor, however, an attention mechanism is used in order to weight the output of the BiLSTM before being used by the CRF layer to compute the probability. The attention mechanism used is shown in Figure 2 using the weight $\alpha = \overline{\sigma(O)}$ where O is the output of a fully connected layer which takes the input of the extractor.

4.4 Morphological Embedding

This architecture is a prefix that can be added to the previous models. It consists of filtering the two words before and after along the one being classified to create a new embedded vector, using one vector for every feature as shown in Figure 4. Therefore, it should be possible to train all the extractors above with a special embedding layer.

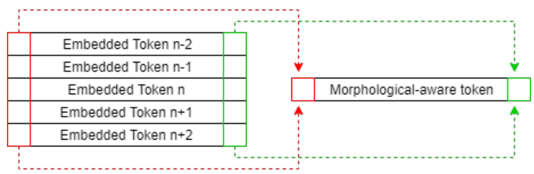

Figure 4: Morphological Embedding

The use of convolutions in textual sequence tagging has shown to [10] increase the performance of the models without a convolutional layer.

5 Experiments

5.1 Set-up

Training

The training of the extractors was done at the extraction element level. In a 3-fold validation fashion, using Adam optimization[8]. Three measurements are taken, Precision (P), Recall(R), and F-score (F). Precision is the fraction of all instances classified correctly as members of the class over all the classified instances of the class. Recall is the fraction of all correctly classified instances over all the instances of the class. The F-score is the harmonic average of the precision and the recall.

$$F = \frac{2(PR)}{P + R}$$

In previous works two main things have been done differently when training and evaluating. Firstly, another embedded vector with information about the word shape is concatenated to the word2vec and POS vectors. The vector contains different values depending on the character types of the original word. Secondly, the true positives(TP) and false positives(FP) were calculated with a length threshold $t \in [0.8, 1.0]$[15]. t would a completeness threshold which determined the minimum amount of element sub-string that was needed to consider a TP. For instance, if $t = 0.8$ and the element to be extracted is a string s and $|s| = 10$, the extraction would be considered a TP if just a substring s_1 is $|s_1| \geq 8$ but it would be considered a FP if $|s_1| < 8$. This method of calculating the TP and FP is not feasible with the encrypted dataset as the length of the strings in previous works was measured at the character level. Therefore, in this case, when evaluating for TP $t = 1$, the extraction will be done at the word level. Therefore an extraction is considered TP if the word extracted is part of the complete element. These measurements allow the performance of each model on every class to be understood and the ability to determine analytically which one performed the best.

Furthermore, it is also possible that different extractors perform better on different super-entities. By super-entity it is meant an entity from the general NER task which are: Organization, Person, Location, Value and Date [5]. Most of the elements to be extracted are sub-entities of the general NER task. Here are the groups of super-entities:

- Date: Start Date, Effective Date, Termination Date
- Organization: Parties, Governing Law

5.2 Results

Performance of Extractors for all elements

Element	avgP	avgR	avgF	varP	varR	varF
CNP	0.623	0.745	0.678	0.000	0.003	0.001
STD	0.583	0.459	0.504	0.002	0.017	0.007
EFD	0.582	0.332	0.412	0.008	0.007	0.002
TED	0.450	0.399	0.399	0.008	0.053	0.032
GOV	0.651	0.841	0.733	0.000	0.007	0.001

Table 1: BiLSTM-LSTM-LR measurements

Element	avgP	avgR	avgF	varP	varR	varF
CNP	0.575	0.648	0.609	0.001	0.005	0.002
STD	0.601	0.432	0.490	0.003	0.039	0.021
EFD	0.616	0.555	0.573	0.000	0.039	0.015
TED	0.437	0.355	0.377	0.000	0.027	0.011
GOV	0.649	0.575	0.600	0.000	0.033	0.014

Table 2: BiLSTM-CRF measurements

Element	avgP	avgR	avgF	varP	varR	varF
CNP	0.587	0.436	0.491	0.000	0.022	0.011
STD	0.595	0.524	0.555	0.002	0.017	0.009
EFD	0.553	0.255	0.349	0.008	0.002	0.004
TED	0.519	0.325	0.382	0.014	0.009	0.002
GOV	0.650	0.584	0.610	0.000	0.017	0.004

Table 3: Att-BiLSTM-CRF measurements

Element	avgP	avgR	avgF	varP	varR	varF
CNP	0.672	0.710	0.689	0.005	0.016	0.008
STD	0.562	0.591	0.575	0.000	0.004	0.001
EFD	0.619	0.543	0.576	0.015	0.034	0.024
TED	0.641	0.564	0.595	0.002	0.008	0.001
GOV	0.699	0.760	0.725	0.000	0.020	0.006

Table 4: Conv-BiLSTM-LSTM-LR measurements

Element	avgP	avgR	avgF	varP	varR	varF
CNP	0.654	0.747	0.697	0.007	0.007	0.007
STD	0.611	0.670	0.628	0.007	0.017	0.000
EFD	0.613	0.645	0.627	0.007	0.000	0.003
TED	0.544	0.513	0.503	0.025	0.030	0.009
GOV	0.723	0.653	0.681	0.000	0.018	0.006

Table 5: Conv-BiLSTM-CRF measurements

Element	avgP	avgR	avgF	varP	varR	varF
CNP	0.662	0.475	0.552	0.007	0.003	0.003
STD	0.625	0.542	0.571	0.007	0.039	0.022
EFD	0.622	0.427	0.503	0.010	0.015	0.013
TED	0.625	0.487	0.527	0.001	0.049	0.026
GOV	0.743	0.632	0.681	0.000	0.009	0.004

Table 6: Conv-Att-BiLSTM-CRF measurements

The tables above show the average P, R and F measurements of a 3-fold validation experiment over the entire dataset where $dropout\_rate = 0.1$ and $batch\_size = 1024$. Additionally, all CRF extractors had a probability threshold $\pi = 0.75$ and the BiLSTM-LSTM-LR extractors had a probability threshold $\pi = 0.5$. The tables above show that in general the extractors extract around one half of the wanted elements. It is also visible that recall decreases with the introduction of the attention mechanism. In the Figure 5 one can see boxplots of the means of F-scores of each extractor for all elements where there seems to be a slightly better performance from the extractors with morphological embedding, and there seems to be no much of an effect with the attention mechanism.

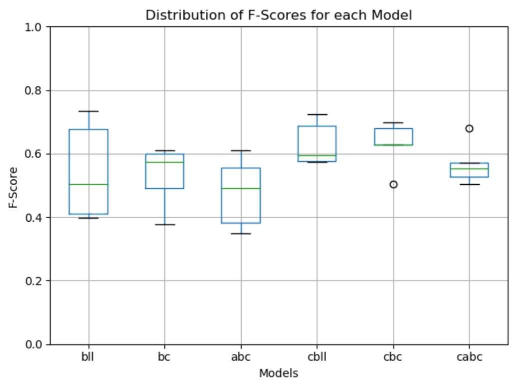

Figure 5: Boxplots of F-scores per extractor

Attention Effect

In Table 7 the results of a paired t-test between the F-scores of extractors with attention mechanism and ones without is shown. The t-test in this scenario has a null hypothesis H_0: Attention mechanisms does not affect the performance of the extractor. For all classes we fail to reject H_0 with a significance level $\alpha = 0.01$.

class	p-value	t-stat
CNP	0.015803285922226947	2.9011187948277266
STD	0.9567403658623226	-0.05561968048435915
EFD	0.016209026361292532	2.8863206847164147
TED	0.8429846691340943	-0.203290918757305
GOV	0.9200846091220802	-0.10288967840834842

Table 7: Results of t-test for independence between extractors with attention mechanism and ones without

Morphological-Embedding Effect

In Table 7 the results of a paired t-test between the F-scores of extractors with morphological embedding and ones without is shown. The t-test in this scenario has a null hypothesis H_0:Morphological embedding doesn't affect the performance of the extractor. For all classes, except for Termination Date (TED) we fail to reject H_0 with a significance level $\alpha = 0.01$.

class	p-value	t-stat
CNP	0.2665235768305611	1.151300127905782
STD	0.10158242537559635	1.737075105609873
EFD	0.0405861593071421	2.227890385976008
TED	0.0061886642927593905	3.150503032132089
GOV	0.2253542640374267	1.2610921677314313

Table 8: Results of t-test for independence between morphological embedding models and ones without

5.3 Performance on Super Entities

Date Sub-Entity In Table 9 the results of a χ^2 goodness of fit test over all extractors for each Date sub-entity is shown. In this test, the null hypothesis H_0:There is no difference in the performance off all extractors over the Date entity sub-classes. For the three classes tested, it is concluded that there is not enough statistical evidence to reject H_0.

class	chi2	p-value
STD	0.023041593611425843	0.9999957483498456
EFD	0.11338131231239851	0.9997788697201235
TED	0.0887775786181445	0.9998789801449095

Table 9: Results of chi-square for independence of extractor performance in Date sub-entities

Organization Sub-Entity In Table 10 the results of a χ^2 goodness of fit test over all extractors for each Organization sub-entity is shown. In this test, the null hypothesis H_0:There is no difference in the performance off all extractors over the Organization entity sub-classes. For the two classes tested, it is concluded that there is not enough statistical evidence to reject H_0.

class	chi2	p-value
CNP	0.05742624033663828	0.9999588162596439
GOV	0.02337440041082657	0.9999955936817989

Table 10: Results of chi-square for independence of extractor performance in Organization sub-entities

6 Discussion

As was shown in the section above, there are two main conclusions that can be derived. The first is that there is statistically significant evidence showing that in one of the elements adding morphological embedding increases the performance of the extractor. The second one is that the extractors trained in this study under-performed compared to the ones in previous works. The latter might be explained by the lack of word shape information in the input data. The lack of shape information could affect the t raining as for some elements like Parties and Governmental Law capitalization of certain characters might indicate their belonging in the class or not. This can also be extended to date elements, since knowing if the characters are numeric or alphabetical can aid in the classification of the element. Moreover, in previous works hyper-parameter tuning was done at the extractor level, where different drop-out rates and batch sizes were tested, while in this set of extractors only one drop-out rate and one batch size were tested.

Furthermore, the results on Table 7 show that there is no statistically significant results to suggest that the addition of the proposed attention mechanism affects the performance of the extractor. In contrast, the results on Table 8 show that the result of morphological embedding extractors can lead to better results.

Moreover, the results of the χ^2 tests where the extractors were tested to see their performance over the super-entities Date and Organization, all conclude on the same results. The results show that no extractors performed significantly better than the other p-values ≈ 1 for all the tests.

The first research question of this paper aims to look for the effect of the introduction of an attention mechanism to the BiLSTM-CRF architecture. As one can see from the results above, there is no significant change of performance in the extractors. However, there is a weak trend of lower recall in the extractors with an attention mechanism.

The second research question of this paper aims to look for the effects of morphological word embedding in the extractors. As it is shown in the results above, there is enough evidence to suggest that morphological embedding can help the performance of an extractor.

The third research question of this paper aims to look is there is any specific architecture that performs better over the super-entities presented above. The results above show that there is no evidence that any model performs better than the other ones over the super-entities.

To conclude, it is shown that the introduction of morphological embedding can help improve the performance of the BiLSTM-LSTM and BiLSTM-CRF extractors. However, further hyper-parameter tuning and the use of information regarding the shape of the word can lead to stronger conclusions. It is proposed that the Conv-BiLSTM-CRF is the best architecture tested in this paper for two main reasons. It outperforms all the other extractors except for the Conv-BiLSTM-LSTM-CRF, and it has less trainable parameters that the BiLSTM-LSTM-CRF, which can help reduce training.

7 Further Research

In order to fully explore the effects of attention mechanisms and morphological embedding in the BiLSTM-CRF extractor it would be necessary to replicate this experiment with different hyper-parameters and the addition of word-shape information. Additionally, a higher k in the k-fold validation process might be useful in order to be able to draw stronger statistical conclusions. This task could be benefited by the idea of meta-learning for super entities and use for fast adaptation when a new element is to be introduced [6] as several elements that can be extracted are part of the same general NER task. Therefore using meta-learning by having pre-trained models can accelerate the training of the extractors.

References

[1] Cao, Kris and Rei, Marek (2016). A joint model for word embedding and word morphology. *arXiv preprint arXiv:1606.02601*.

[2] Chalkidis, Ilias, Androutsopoulos, Ion, and Michos, Achilleas (2017). Extracting contract elements. *Proc. of the 16th Int. Conf. on Artificial Intelligence and Law*, pp. 19–28.

[3] Chang, Angel X and Manning, Christopher D (2014). Tokensregex: Defining cascaded regular expressions over tokens. *Tech. Rep. CSTR 2014-02*.

[4] Cotterell, Ryan and Heigold, Georg (2017). Cross-lingual, character-level neural morphological tagging. *arXiv preprint arXiv:1708.09157*.

[5] Dernoncourt, Franck, Lee, Ji Young, and Szolovits, Peter (2017). Neuroner: an easy-to-use program for named-entity recognition based on neural networks. *arXiv preprint arXiv:1705.05487*.

[6] Finn, Chelsea, Abbeel, Pieter, and Levine, Sergey (2017). Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*.

[7] Hochreiter, Sepp and Schmidhuber, Jürgen (1997). Long short-term memory. *Neural computation*, Vol. 9, No. 8, pp. 1735–1780.

[8] Kingma, Diederik P and Ba, Jimmy (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[9] Luo, Ling, Yang, Zhihao, Yang, Pei, Zhang, Yin, Wang, Lei, Lin, Hongfei, and Wang, Jian (2017). An attention-based bilstm-crf approach to document-level chemical named entity recognition. *Bioinformatics*, Vol. 1, p. 8.

[10] Ma, Xuezhe and Hovy, Eduard (2016). End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

[11] Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, pp. 3111–3119.

[12] Olah, Chris and Carter, Shan (2016). Attention and augmented recurrent neural networks. *Distill*, Vol. 1, No. 9, p. e1.

[13] Shen, Sheng-syun and Lee, Hung-yi (2016). Neural attention models for sequence classification: Analysis and application to key term extraction and dialogue act detection. *arXiv preprint arXiv:1604.00077*.

[14] Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan (2014).

Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, Vol. 15, No. 1, pp. 1929–1958.

[15] Wyner, A and Casini, G (2017). A deep learning approach to contract element extraction. *Legal Knowledge and Information Systems*, p. 155.

[16] Zheng, Shuai, Jayasumana, Sadeep, Romera-Paredes, Bernardino, Vineet, Vibhav, Su, Zhizhong, Du, Dalong, Huang, Chang, and Torr, Philip HS (2015). Conditional random fields as recurrent neural networks. *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1529–1537.

Appendices

Tables

	model	P	R	F
0	bll	0.6126	0.7848	0.6881
1	bc	0.5435	0.568	0.5555
2	abc	0.5879	0.2756	0.3753
3	cbll	0.5977	0.5811	0.5893
4	cbc	0.6296	0.767	0.6915
5	cabc	0.6715	0.533	0.5943
6	bll	0.64	0.7717	0.6997
7	bc	0.5885	0.7048	0.6414
8	abc	0.5943	0.5639	0.5787
9	cbll	0.7301	0.7139	0.7219
10	cbc	0.7473	0.8173	0.7807
11	cabc	0.5756	0.4331	0.4943
12	bll	0.6168	0.6779	0.6459
13	bc	0.594	0.6702	0.6298
14	abc	0.5779	0.4689	0.5178
15	cbll	0.6895	0.8356	0.7555
16	cbc	0.5858	0.6563	0.6191
17	cabc	0.7383	0.4602	0.567

Table 11: All measurements gathered from element Parties

	model	P	R	F
0	bll	0.6064	0.5404	0.5715
1	bc	0.6571	0.6508	0.6539
2	abc	0.5613	0.4341	0.4896
3	cbll	0.5603	0.6415	0.5981
4	cbc	0.6464	0.5764	0.6094
5	cabc	0.6069	0.6763	0.6397
6	bll	0.535	0.529	0.532
7	bc	0.5475	0.3734	0.444
8	abc	0.5773	0.4634	0.5141
9	cbll	0.5679	0.6162	0.5911
10	cbc	0.6686	0.6151	0.6407
11	cabc	0.7134	0.6353	0.6721
12	bll	0.6069	0.3066	0.4074
13	bc	0.5986	0.2714	0.3734
14	abc	0.6462	0.6758	0.6606
15	cbll	0.5568	0.5155	0.5353
16	cbc	0.5173	0.8174	0.6336
17	cabc	0.554	0.3143	0.4011

Table 12: All measurements gathered from element Starting Date

	model	P	R	F
0	bll	0.6799	0.2559	0.3718
1	bc	0.6249	0.5922	0.6081
2	abc	0.6572	0.297	0.4091
3	cbll	0.7571	0.7558	0.7565
4	cbc	0.6313	0.6418	0.6365
5	cabc	0.5149	0.3596	0.4235
6	bll	0.567	0.3178	0.4073
7	bc	0.6276	0.7312	0.6755
8	abc	0.52	0.2626	0.349
9	cbll	0.5784	0.4214	0.4876
10	cbc	0.689	0.6548	0.6715
11	cabc	0.7121	0.5686	0.6323
12	bll	0.4983	0.4212	0.4565
13	bc	0.5963	0.3423	0.435
14	abc	0.483	0.2066	0.2895
15	cbll	0.5228	0.452	0.4848
16	cbc	0.5196	0.6369	0.5723
17	cabc	0.6393	0.3525	0.4544

Table 13: All measurements gathered from element Effective Date

	model	P	R	F
0	bll	0.5532	0.5311	0.5419
1	bc	0.426	0.1873	0.2602
2	abc	0.5467	0.32	0.4037
3	cbll	0.6642	0.5453	0.5989
4	cbc	0.7075	0.5227	0.6012
5	cabc	0.6252	0.2397	0.3466
6	bll	0.3998	0.532	0.4565
7	bc	0.4612	0.3623	0.4058
8	abc	0.6199	0.2311	0.3367
9	cbll	0.5912	0.6589	0.6232
10	cbc	0.3925	0.6808	0.4979
11	cabc	0.5999	0.5544	0.5763
12	bll	0.3957	0.133	0.1991
13	bc	0.4253	0.5157	0.4662
14	abc	0.3904	0.4239	0.4065
15	cbll	0.6673	0.4883	0.5639
16	cbc	0.5324	0.335	0.4112
17	cabc	0.6509	0.6681	0.6594

Table 14: All measurements gathered from element Termination Date

	model	P	R	F
0	bll	0.6434	0.8363	0.7273
1	bc	0.6398	0.3684	0.4676
2	abc	0.6396	0.7348	0.6839
3	cbll	0.7125	0.8322	0.7677
4	cbc	0.7274	0.7868	0.756
5	cabc	0.7594	0.737	0.748
6	bll	0.6533	0.7606	0.7029
7	bc	0.666	0.7074	0.6861
8	abc	0.6491	0.5079	0.5699
9	cbll	0.6776	0.5973	0.6349
10	cbc	0.7389	0.5152	0.6071
11	cabc	0.733	0.5614	0.6358
12	bll	0.6557	0.9275	0.7682
13	bc	0.6426	0.6494	0.646
14	abc	0.6604	0.5102	0.5757
15	cbll	0.706	0.85	0.7713
16	cbc	0.703	0.658	0.6797
17	cabc	0.7359	0.597	0.6592

Table 15: All measurements gathered from element Governmental Law