Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Special Track on AI in FinTech

Deep Semantic Compliance Advisor for Unstructured Document Compliance Checking

Honglei Guo*, **Bang An**, **Zhili Guo** and **Zhong Su**
IBM Research - China
{guohl, abangbj, guozhili, suzhong}@cn.ibm.com

Abstract

Unstructured document compliance checking is always a big challenge for banks since huge amounts of contracts and regulations written in natural language require professionals' interpretation and judgment. Traditional rule-based or keyword-based methods cannot precisely characterize the deep semantic distribution in the unstructured document semantic compliance checking due to the semantic complexity of contracts and regulations. Deep Semantic Compliance Advisor (DSCA) is an unstructured document compliance checking platform which provides multi-level semantic comparison by deep learning algorithms. In the statement-level semantic comparison, a Graph Neural Network (GNN) based syntactic sentence encoder is proposed to capture the complicate syntactic and semantic clues of the statement sentences. This GNN-based encoder outperforms existing syntactic sentence encoders in deep semantic comparison and is more beneficial for long sentences. In the clause-level semantic comparison, an attention-based semantic relatedness detection model is applied to find the most relevant legal clauses. DSCA significantly enhances the productivity of legal professionals in the unstructured document compliance checking for banks.

1 Introduction

Governments around the world have tightened regulatory controls and placed substantially significant penalties on noncompliance in recent years. Banks have to spend billions on regulatory compliance and litigation each year. Banks need solutions to quickly assess, manage, and maintain their business compliance and reduce the noncompliance risks. However, most of the compliance management solutions provide compliance checking on the structured data and business rules[Butler and OBrien, 2018]. They can't provide regulatory compliance checking on contracts and regulations. Regulatory compliance against these unstructured legal documents is still a big challenge for banks since it is a high-cost, low-efficient and inconsistent process due to the manual review required.

*Contact Author

We developed Deep Semantic Compliance Advisor (DSCA) for solving the above challenges. Different from the existing solutions, DSCA provides deep semantic compliance checking for contracts and regulations. For example, all the contracts are analyzed by DSCA collectively. A smart checking engine would look at every contract against the criteria repository and highlight the dubious or incomplete contract clauses for legal professionals to double check. By leveraging document understanding and multi-level semantic comparison, DSCA can identify the most relevant reference clauses for the legal clauses under review. The legal professionals may further view the detailed semantic comparison of the relevant clause pairs with differences clearly highlighted. They can quickly figure out which clauses they could accept or counter.

Traditional rule-based or keyword-based methods can not effectively characterize the deep semantic distribution of the contracts and regulations due to their semantic complexity. We attempt to employ deep learning technology to capture both explicit and hidden semantic association in the semantic comparison. In the statement-level deep semantic comparison, we propose a GNN-based sentence encoder to combine both syntactic information and context semantics of the statement sentences, which outperforms existing syntactic sentence encoders in our experiment. In the clause-level semantic comparison, one big challenge is to find the semantic relevant clause pairs between the contract under review and the reference one since the sequence of the clauses and wording style are often different. Hence, we propose an attention-based deep semantic relatedness detection model to automatically find the most relevant clause pairs according to topic semantic relatedness of clauses. Experimental results show that our model outperforms both Support Vector Machine (SVM) algorithm and Long Short-Term Memory(LSTM) representation model.

In sum, the key technical advantages of DSCA are below.

- Deep statement semantic comparison with GNN-based sentence encoder
- Clause semantic relatedness detection with neural attention model.

2 Related Work

Deep semantic comparison is a big challenge in unstructured document compliance checking. It focuses on checking the semantic relations between the statement sentences, including

entailment, contradiction and neutral. Such semantic comparison actually is a natural language inference (NLI) task.

Syntactic information can provide rich semantic association clues for natural language inference. There are three representative syntactic sentence encoders in previous works on NLI. Tree-LSTM [Tai *et al.*, 2015] [Chen *et al.*, 2017] is a recursive network that composes the meaning of sub-trees according to the syntax with expensive recursion computation. Tree-based Convolutional Neural Network (CNN) [Mou *et al.*, 2016] does convolutions on the dependency parse tree and is shown to be more robust than sequential convolution in terms of word order distortion. Phrase-level self-attention network (PSAN) [Wu *et al.*, 2018] utilizes a constituency parse tree and hierarchically captures context dependencies at the phrase level instead of the sentence level. Our model uses GNNs to make utilizing the syntactic structure more flexible.

Recurrent neural networks(RNNs) have been used to improve language model [Mikolov *et al.*, 2010] and sentence embedding [Palangi *et al.*, 2015]. Although a great success in a variety of tasks [Bengio *et al.*, 1994; Hochreiter, 1998], RNNs are not good at memorizing long or distance sequences [Sutskever *et al.*, 2014]. In order to improve the result in translating longer sentences, [Bahdanau *et al.*, 2014] propose to use attention in RNN models, which allows RNNs to selectively focus on the most task-relevant parts of input sequence, assign importance weights to those parts and join them into a single representation. Our clause semantic relatedness detection method is motivated by the central role of the neural attention mechanism in machine translation [Bahdanau *et al.*, 2014]. Since legal clause often consists of long text snippet, we may leverage an attention-based model to find the most relevant reference clauses for the given clauses.

3 Semantic Compliance Advisor Overview

DSCA is a semantic compliance advisor platform for unstructured document compliance checking (see Figure 1). DSCA can provide contract compliance, regulation tracking, cross-region compliance applications by leveraging document ingestion and deep semantic comparison models. We take contract compliance checking as an example to illustrate the key components of DSCA in this paper.

In the document ingestion phase, clause parser extracts legal clauses from the contract using patterns and rules. In the semantic comparison phase, DSCA provides multi-level semantic comparison, including document-level, clause-level and statement-level comparison. In the statement-level semantic comparison, in order to effectively capture the underlying semantic association among the statement sentences, we build a deep statement semantic comparison model which encodes syntactic information and sequence context information using GNN-based sentence encoder. More details will be illustrated in Section 4. In the clause-level semantic comparison, we propose an attention-based clause semantic relatedness detection model to automatically find the most relevant clause pairs. We employ word by word attention mechanism and two LSTM encoders to calculate attention weights among the words and the clauses for the semantic relatedness comparison. Then an output LSTM layer is used to summarize the comparison

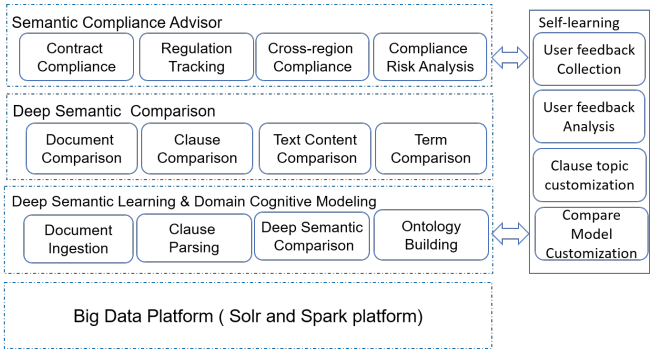

Figure 1: Deep Semantic Compliance Advisor (DSCA) overview

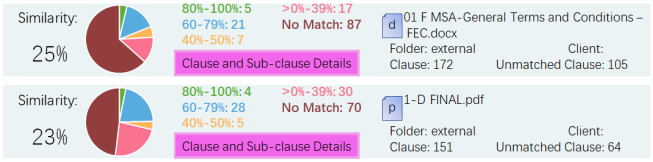

Figure 2: Document level comparison

results and output the final label (i.e. relevant or irrelevant). More details will be illustrated in Section 5. In the document-level semantic comparison, DSCA provides an overview of the similarity distribution of the most relevant clause pairs (see Figure 2). The reviewer may look into the detail statement semantic comparison of each relevant clause pair with differences clearly highlighted. If the clauses in the contract under review are not mentioned in the reference one, the reviewer should check them carefully.

By leveraging deep semantic comparison, DSCA improves the productivity of legal professionals in the unstructured document compliance checking.

4 GNN-based Deep Statement Semantic Comparison

Deep semantic comparison is the key task in unstructured document compliance checking. It focuses on detecting the semantic inferential relationship (i.e. *entailment*, *contradiction* or *neutral*) between the special statements in the contracts under review and the standard ones in the reference contracts or regulations. The key challenge for deep semantic comparison is how to capture the semantic association in statements, especially for long legal statements. We propose a deep semantic comparison framework which employs GNN-based sentence encoder to effectively encode the syntactic and semantic association among statements.

4.1 Deep Statement Semantic Comparison Framework

Deep statement semantic comparison in compliance checking actually is a natural language inference (NLI) task. If the given statement (a premise) semantically entails the standard statement (a hypothesis), the given statement is considered as compliance. If they are contradiction, the given statement is considered as noncompliance.

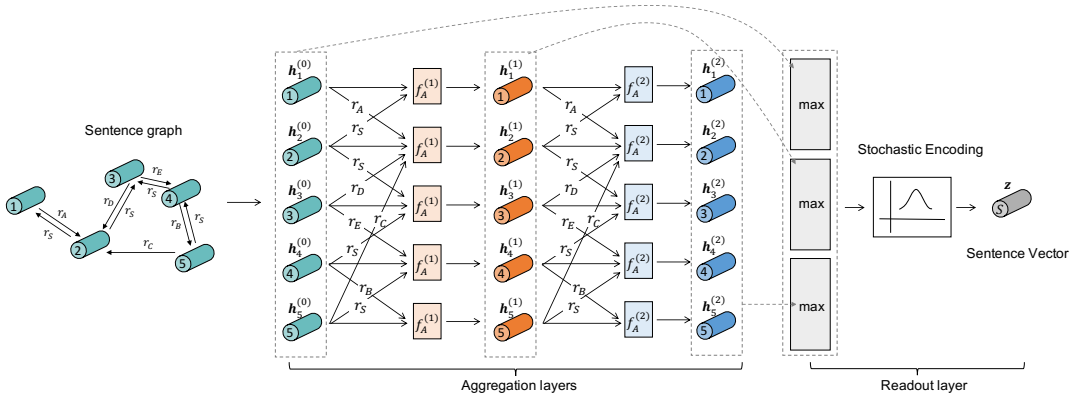

Figure 3: An illustration of the proposed syntactic sentence encoder. A cylinder indicates a word vector or a sentence vector. r_i indicates a relation type. $\boldsymbol{h}_j^{(i)}$ indicates the latent representation of the j-th token in i-th layer. $f_A^{(i)}$ indicates the aggregation function in i-th layer. The semantic of the sentence is encoded in a vector after the aggregation layers and the readout layer.

Generally, NLI models follow the *Siamese* architecture [Bromley *et al.*, 1994] to encode premise and hypothesis separately through the same sentence encoder and then do classification based on two sentence vectors. Existing sentence encoders in NLI typically utilize RNNs and self-attention mechanism [Bowman *et al.*, 2015; Mou *et al.*, 2016; Conneau *et al.*, 2017; Im and Cho, 2017; Yoon *et al.*, 2018]. However, RNNs are hard to parallelize or capture long-term dependencies. Attentions only rely on word representations, which may suffer from syntax variants and the required memory grows quadratically with the sentence length. The key for statement semantic comparison is to learn an efficient sentence encoder with strong semantic capturing ability. To achieve that, we leverage syntactic structure of sentences as a prior knowledge since it provides us rich semantic clues for understanding sentences. The existing syntactic sentence encoders encode the sentence structure as a tree, which loses flexibility and limits the expressiveness of the model. Hence, we propose a novel GNN-based syntactic sentence encoder for semantic comparison task.

In our framework, a sentence is first converted into a graph, then encoded by the proposed GNN-based syntactic sentence encoder (see section 4.2). With the intrinsic advantages of GNN, our model allows contextual messages propagating along the sentence structure which is more flexible and efficient in incorporating syntax-aware latent representations. Besides, our model is well parallelized without recursion computation. At last, the comparison model inference on two sentence vectors captured by GNN-based encoder (see section 4.3).

4.2 GNN-based Syntactic Sentence Encoder

Our syntactic sentence encoder (see Figure 3) contains two types of layers. Specifically, the aggregation layer enables context propagates along sentence structures to encode syntax-aware contextual representation of each token. The readout layer employs the stochastic encoding mechanism to obtain the sentence embedding with better generalization ability.

Sentence graph generation. As the input of the GNN-based encoder, a sentence is represented by a labeled directed asymmetric sentence graph. The sentence graph contains both sequential and syntactic structures. Given a sentence, node v_i represents the i-th token in sentence. Each edge is a tuple $e_{ij} = (v_i, v_j, r_{ij})$, where r_{ij} is the relation label from word v_j to word v_i. In sentence graph, there are three types of relations: *dependency*, *opposite dependency*, *sequence*. Dependency relation (v_i, v_j, d_{ij}) indicates a grammatical relation $d_{ij} \in D_r$, *e.g.'nsubj'*, holds between the governor v_i and the dependent v_j. D_r is the set of all dependency relations. Opposite dependency relation is the opposite of that with an opposite labeled type, *e.g.'op-nsubj'*. Sequence relations $(v_i, v_{i+1}, \textit{'seq'})$ and $(v_i, v_{i-1}, \textit{'seq'})$ with the type label *'seq'* are also added into the sentence graph. If there are more than one relation from one node to another, we will keep the dependency relation. In the semantic comparison, we use $\boldsymbol{X} \in \mathbb{R}^{N \times d_0}$ as the input word embedding matrix, $\boldsymbol{H} \in \mathbb{R}^{N \times d}$ as the latent node feature matrix and $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ as the labeled adjacency matrix, where N is the number of nodes and d is the dimension of node features. $\boldsymbol{H}$ and $\boldsymbol{A}$ together depict the sentence graph.

Aggregation layer. In this layer, every node aggregates messages from its neighbors that have direct structural connections with it. In our model, the context messages of node v_i in k-th layer are aggregated, with a softmax function, to $\boldsymbol{c}_i^{(k)}$ according to different relation types between it and its neighbors $\mathcal{N}(v_i)$ as

$$\boldsymbol{c}_i^{(k)} = \sum_{v_j \in \mathcal{N}(v_i)} \left[\frac{\exp\left(a_{ij}^{(k)}\right)}{\sum_{v_j \in \mathcal{N}(v_i)} \exp\left(a_{ij}^{(k)}\right)} \boldsymbol{h}_j^{(k-1)} \right]$$

where $\boldsymbol{h}_j^{(k-1)}$ is the latent vector of node v_j, $a_{ij}^{(k)}$ is the learnable weight parameter of the relation r_{ij} in the k-th layer indicating the contribution of neighbor v_j. Combining the context message together with its feature vector and then passing through a multi-layer perception (MLP) leads to the syntactic-context-aware representation of node v_i as

$$\boldsymbol{h}_i^{(k)} = \mathsf{MLP}^{(k)} \left[\epsilon^{(k)} \boldsymbol{h}_i^{(k-1)} + (1 - \epsilon^{(k)}) \boldsymbol{c}_i^{(k)} \right]$$

or matrix form for whole graph as

$$\boldsymbol{H}^{(k)} = \mathsf{MLP}^{(k)} \left[\epsilon^{(k)} \boldsymbol{H}^{(k-1)} + (1 - \epsilon^{(k)}) \boldsymbol{A}_n^{(k)} \boldsymbol{H}^{(k-1)} \right]$$

where $\epsilon^{(k)}$ is a global learnable parameter, $\boldsymbol{A}_n^{(k)} \in \mathbb{R}^{N\times N}$ is the relation matrix after softmax. By stacking multiple aggregation layers, we would get the feature vector of every token containing contexts from its large-hop neighbors.

Readout layer. In order to get the sentence representation after aggregation layers, we first conduct a readout function as

$$\boldsymbol{h}_S = \mathsf{max}\left[\mathsf{concat}\left(\boldsymbol{H}^{(0)}, \boldsymbol{H}^{(1)}, \boldsymbol{H}^{(2)}\right)\right]$$

where we assume there are 2 aggregation layers and the sentence vector $\boldsymbol{h}_S \in \mathbb{R}^{d^{(0)}+d^{(1)}+d^{(2)}}$ is derived from the max-pooling on every dimension after the concatenation of every aggregation layer's output. With hierarchical max-pooling, the sentence vector $\boldsymbol{h}_S$ contains rich information but may also suffer redundancy at the same time. In order to eliminate the redundancy and improve the generalization ability at the same time, we employ a *stochastic encoding module* [Alemi *et al.*, 2017] to encode the sentence representation $\boldsymbol{h}_S$ into the stochastic sentence representation $\boldsymbol{z}$ as

$$\boldsymbol{z} \sim p(\boldsymbol{z}|\boldsymbol{h}_S)$$
$$p(\boldsymbol{z}|\boldsymbol{h}_S) = \mathcal{N}(\boldsymbol{z}; L_\mu(\boldsymbol{h}_S), L_\sigma(\boldsymbol{h}_S)\boldsymbol{I})$$

where $\boldsymbol{z}$ is sampled from the posterior $p(\boldsymbol{z}|\boldsymbol{h}_S)$ which is a multivariate Gaussian distribution with diagonal covariance matrix whose parameters are computed from $\boldsymbol{h}_S$. L_μ and L_σ are two linear layers. By controlling mutual information between $\boldsymbol{z}$ and $\boldsymbol{h}_S$ when learning, we would get the sentence representation $\boldsymbol{z}$ with lower dimension but higher generalization ability guaranteed by previous works [Xu and Raginsky, 2017] and confirmed by our experiments.

4.3 Deep Semantic Comparison Model

Through the GNN-based sentence encoder, the premise and the hypothesis are encoded into two sentence vectors (we use $\boldsymbol{p}$ and $\boldsymbol{h}$ to denote them respectively). To inference the semantic relation of the premise and the hypothesis, a matching layer $\boldsymbol{m} = [\boldsymbol{p}; \boldsymbol{h}; |\boldsymbol{p} - \boldsymbol{h}|; \boldsymbol{p} \odot \boldsymbol{h}]$ [Mou *et al.*, 2016] is applied on the top of the encoder and the entailment is done by a classifier on it. Note that, the loss function in our approach is a combination of the cross entropy and the regularization on mutual information between $\boldsymbol{z}$ and $\boldsymbol{h}_S$.

In the model learning, the pre-trained word embedding GloVe (GloVe 840B 300D) [Pennington *et al.*, 2014] is employed and is fixed during training. We use universal dependency parser conducted by StanfordNLP [Qi *et al.*, 2018] to generate dependency relations. 2-layer MLP is applied in aggregation layers and 2-layer MLP is applied for classification. ReLU is used as the activation function. Dropout with rate 0.1 is applied after each MLP layer (except the last layer). β is set to 1e-6. Learning rate is initialized as 5e-4 and is decreased by the factor of 0.2 if the performance does not improve after an epoch. We use Adam [Kingma and Ba, 2015] as the optimizer and set batch size to 64. Mean value of $p(\boldsymbol{z}|\boldsymbol{x})$ rather than the sampled ones is used as the sentence representation during inference. Models are evaluated on the validation data after each epoch and early stop with 3 epoch patience.

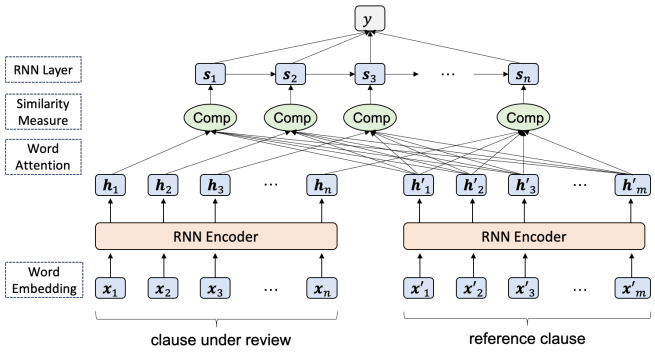

Figure 4: The architecture of neural attention model

5 Attention-based Clause Semantic Relatedness Detection

In the clause-level semantic comparison, it is very challenging to find the semantic relevant clause pairs since the sequence of the clauses and wording style are often varied with the documents. DSCA employs clause semantic relatedness detection to find the most relevant clause pairs between the contract under review and the reference one. We consider this task as a learning-based binary classification task. Given two contract clauses, the model will give a label (i.e. relevant or irrelevant).

5.1 Neural Attention Model for Clause Semantic Relatedness Detection

The core model consists of the following three components(see Figure 4), which are trained jointly: 1) word by word attention, 2) compare measurement and 3) RNN output layers.

Word by word attention. Attentive neural networks have recently demonstrated success in a wide range of tasks such as machine translation [Bahdanau *et al.*, 2014], image captioning [Xu *et al.*, 2015] and sentence inference [Rocktäschel *et al.*, 2016]. The idea is to allow the model to attend over past output vectors. In the clause semantic relatedness detection, we use word by word attention mechanism to soft align words in the clause under review with the reference one. Two LSTMs are employed to encode the two clauses respectively. Then each output state of the first LSTM attends the second LSTM's output vector. Attention weights α_t and comparisons are calculated over all output vectors of the reference clause for each output state $\boldsymbol{h}_t$ in the clause under review. This can be modeled as follows:

$$\boldsymbol{h}_k^a = \left[\mathsf{Comp}(\Sigma_{j=1}^m \alpha_{kj}\boldsymbol{h}'_j, \boldsymbol{h}_k), \Sigma_{j=1}^m \alpha_{kj}\mathsf{Comp}(\boldsymbol{h}'_j, \boldsymbol{h}_k)\right]$$

$$\text{where, } \alpha_{kj} = \frac{exp(e_{ij})}{\Sigma_{j'} exp(e_{kj'})}$$

$$e_{kj} = \boldsymbol{W}^e \cdot \mathsf{tanh}(\boldsymbol{W}^s\boldsymbol{h}'_j + \boldsymbol{W}^t\boldsymbol{h}_k + \boldsymbol{W}^c\mathsf{Comp}(\boldsymbol{h}'_j, \boldsymbol{h}_k))$$

where $\boldsymbol{h}_k^a$ is the compare result of output state $\boldsymbol{h}_k$ of the clause under review and the output vectors of reference $\boldsymbol{h}'$, all matrices $\boldsymbol{W}$ contain weights to be learned and the function Comp is the compare function detailed in the next section. Note that the compare result $\boldsymbol{h}_k^a$ is an concatenation of two parts: 1) compare result of $\boldsymbol{h}_k$ and an aggregate of vector $\boldsymbol{h}'$ with the

attention weights; 2) an aggregate of compare result of $\boldsymbol{h}_k$ and each reference state $\boldsymbol{h}'_j$. We also assume the soft align weight α_{kj} is relevant to the comparison of $\boldsymbol{h}_k$ and $\boldsymbol{h}'_j$.

Compare measurement. We define the compare function for two states as follows:

$$\mathsf{Comp}(\boldsymbol{x}, \boldsymbol{y}) = [cos(\boldsymbol{x}, \boldsymbol{y}), d_{l_2}(\boldsymbol{x}, \boldsymbol{y}), |\boldsymbol{x} - \boldsymbol{y}|, \boldsymbol{x} \odot \boldsymbol{y}]$$

where cosine distance (first term) and element-wise multiplication (fourth term) measure the distance of two vectors according to the angle between them, Euclidean distance (second term) and element-wise absolute distance (third term) measure magnitude differences. In another perspective, cosine distance and Euclidean distance measure the sum distance and element-wise multiplication and absolute distance measure the element-wise distance in each dimension of the vectors.

RNN output layers. After word by word attention, each output state of the clause under review has a compare result with the reference one and then another LSTM is used to sequentially summarize the compare results. We take the last state of the LSTM as the compare result representation. On top of the LSTM layer, we use a linear layer and a log-softmax layer as the final output layer, which outputs the label (i.e. relevant or irrelevant).

Network learning. To train our models, we use stochastic gradient descent (SGD) to minimize the negative log likelihood loss function and the back-propagation algorithm to compute the gradients. For the output layer, we employ dropout with a constraint on l_2-norm of the weight vectors. The dropout rate is 0.2 and the l_2 constraints is 3. Training is done through stochastic gradient descent over mini-batches with the size of 50 and Adadelta update rule [Zeiler, 2012] and the number of epochs is set to 200. The encoding LSTM layer in DSCA is bidirectional LSTM and the numbers of cell are all set as 20. The hidden state in the output linear layer is set as 50. In our experiment, we use initialized randomly word vectors for each word and learn them as parameters during training. The dimension of word vectors is set to 50.

6 Experiments and Use Case

We evaluate our method on both banking data and available public data in this section. In order to verify the wide applications of our GNN-based sentence encoder, we evaluate our method and the state-of-the-art methods on the available public data set. Experimental results show that our method outperforms other syntactic sentence encoders. Meanwhile, we also evaluate our clause relatedness detection method on a real banking data set from our customer. All the experimental results show that our method achieves better performances in various applications.

6.1 Evaluation on GNN-based Deep Statement Semantic Comparison

In order to compare our GNN-based deep semantic comparison model with existing syntactic sentence encoders and GNN models, we conduct experiments on the public Stanford Natural Language Inference (SNLI) dataset (https://nlp.stanford.edu/projects/snli/).

Model	Acc(%)
Gumble TreeLSTM [Choi *et al.*, 2018]	86.0
TBCNN [Mou *et al.*, 2016]	82.1
PSAN [Wu *et al.*, 2018]	86.1
GCN [Kipf and Welling, 2017]	83.1
GAT [Veličković *et al.*, 2018]	84.0
GGS-NNs [Li *et al.*, 2016]	85.3
Ours	**86.6**

Table 1: Test accuracy (%) of syntactic sentence encoders evaluated on the SNLI dataset.

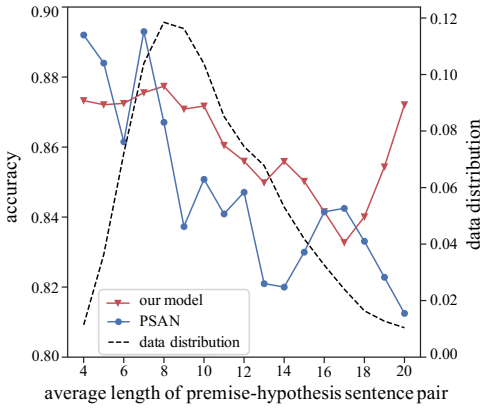

Figure 5: Test accuracy of our model and PSAN on SNLI dataset with respect to different sentence lengths.

Baselines. Gumble TreeLSTM [Choi *et al.*, 2018], TBCNN [Mou *et al.*, 2016], PSAN [Wu *et al.*, 2018] are three representative syntactic sentence encoders. GCN [Kipf and Welling, 2017] is a GNN variant that does convolution on the graph. GAT [Veličković *et al.*, 2018] aggregates messages with attention mechanism. GGS-NNs [Li *et al.*, 2016] uses gated recurrent units for aggregation. All models follow the same Siamese architecture.

Experimental results. As shown in Table 1, our model outperforms all three existing syntactic sentence encoders and three Graph Neural Networks. Furthermore, we evaluate the performance of our model on different length conditions of the premise-hypothesis sentence pair. It is hard to capture the semantic of long sentences due to complex syntax and long-term dependencies. We compare our model with PSAN, the state-of-the-art syntactic sentence encoder which utilizes phrase-level self-attention on constituency parse tree. Figure 5 shows the test accuracy of two models according to different average lengths of premise-hypothesis sentence pairs. As expected, by leveraging dependency relations directly and encoding the sentence as a graph, our model has remarkable advantages on capturing long-term dependencies and extracting semantic meanings of complex sentences.

6.2 Evaluation on Clause Semantic Relatedness Detection

Data set. As there is not any available public data set for evaluating contract clause semantic relatedness, we conduct

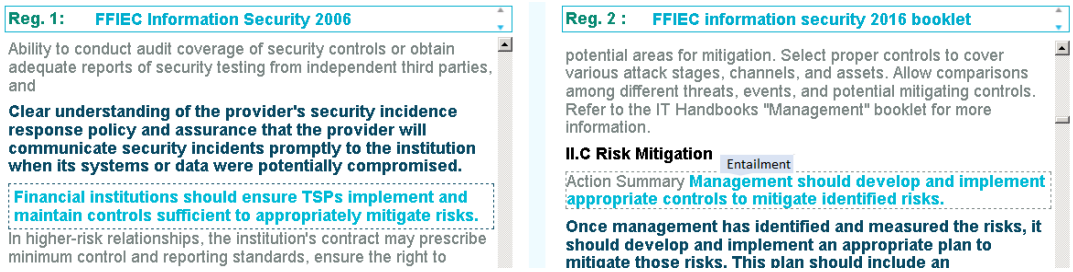

Figure 6: Deep statement semantic comparison

Model	SVM	LSTM	**Ours**
Acc (%)	80.6	89.1	**91.1**

Table 2: Accuracy of clause semantic relatedness detection

experiments on a real English contract data set from our customer. Two legal professionals manually extracted the most relevant clause pairs from the contracts as positive samples. We also randomly construct negative samples with irrelevant clauses in the difference contracts. We use 2,028 clause pairs as training set, 500 pairs as the develop set and 400 pairs as test set. In each set, the positive-negative ratio is about 1:1.

Baselines. In this experiment, our attention-based semantic relatedness detection method is compared with the popular text classification algorithms SVM (Support Vector Machine) and LSTM (Long Short-Term Memory). SVM-based method uses TF-IDF of words to represent the clause and the output of compare function defined in Section 5 as features. LSTM-based method uses two LSTMs to represent the given two clauses respectively, a compare layer of the function defined in Section 5, and a log-softmax layer to give the label.

Experimental results. Experimental results are shown in Table 2. We have the following observations: 1) The models with RNN (Recurrent Neural Network) encoder (LSTM and our neural attention models) perform better than SVM. This shows the effectiveness of learning representation by RNN encoder. 2) Our proposed neural attention model performs better than LSTM model. This shows the effectiveness of learning word soft-alignment by attention models. Pairwise comparison by attention mechanism are relatively more important than global-level representations.

6.3 Use Case

DSCA has been applied for unstructured document compliance checking in one of the largest financial institutions in the world, which operates in over 100 countries and regions around the globe. The key pain-point in this case is that manual processing of a large collection of regulation documents and assessing compliance against them become difficult to manage and scale. DSCA significantly increase the processing speed and consistency in parsing regulations, assessing compliance, and remediate non-compliance issues.

When the reviewer uploads a contract or regulation to DSCA, DSCA analyzes the document to extract all the clauses at first. Then DSCA finds the most relevant clauses in the reference repository using clause semantic relatedness detection model. Finally, DSCA provides multi-level semantic comparison for the reviewer to check the consistency and inconsistency between the document under review and the most relevant reference one. Since customer contracts are confidential, they can not be shown here. Hence, we take two public financial regulations as examples. Figure 6 shows the deep statement semantic comparison between the regulation under review (in the left side) and the reference one (in the right side). The reviewer can easily check how each clause under review semantically matches or differs from the standard clause in the reference document. As shown, DSCA figures out that the statement pair in the left and right dotted boxes has entailment relationship. When the statements in the left side are irrelevant to any statements in the right side, they are grayed. Obviously, the reviewer can quickly figure out which clauses they could accept or counter by multi-level semantic comparison.

Generally, it takes one legal professional 4+ hours for each contract checking while DSCA can return the checking results with detail comparison information in one minute. This significantly saves legal professionals' efforts and time in the compliance checking.

7 Conclusion

Unstructured document compliance checking is a challenging task with high interest in the financial area (particularly for in-house legal and compliance division). The big challenge for unstructured document compliance checking is deep semantic comparison among the contracts and regulations. Our deep semantic compliance advisor provides GNN-based deep statement semantic comparison and attention-based clause semantic relatedness detection for unstructured document compliance checking. The proposed novel deep statement semantic comparison method employs GNN-based syntactic sentence encoder to encode both syntactic information and the surrounding sequence context in the statements. Experimental results indicate that our GNN-based syntactic sentence encoder outperforms the existing syntactic sentence encoders. Our GNN-based model is good at capturing semantic meaning of long sentences and structure information. DSCA platform proposed in this paper can significantly enhance the productivity of legal professionals by leveraging deep learning algorithms. It reduces human efforts and time in the unstructured document compliance checking. Moreover, it enhances the coverage of compliance checking and reduces the potential risks.

References

[Alemi *et al.*, 2017] Alex Alemi, Ian Fischer, Josh Dillon, and Kevin Murphy. Deep variational information bottleneck. In *ICLR*, 2017.

[Bahdanau *et al.*, 2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[Bengio *et al.*, 1994] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[Bowman *et al.*, 2015] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A large annotated corpus for learning natural language inference. In *EMNLP*, pages 632–642, 2015.

[Bromley *et al.*, 1994] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *NIPS*, pages 737–744, 1994.

[Butler and OBrien, 2018] Tom Butler and Leona OBrien. Understanding regtech for digital regulatory compliance. *FinTech and Strategy in the 21st Century*, 2018.

[Chen *et al.*, 2017] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced LSTM for natural language inference. In *ACL*, pages 1657–1668, 2017.

[Choi *et al.*, 2018] Jihun Choi, Kang Min Yoo, and Sang-goo Lee. Learning to compose task-specific tree structures. In *AAAI*, pages 5094–5101, 2018.

[Conneau *et al.*, 2017] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, pages 670–680, 2017.

[Hochreiter, 1998] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

[Im and Cho, 2017] Jinbae Im and Sungzoon Cho. Distance-based self-attention network for natural language inference. *arXiv preprint arXiv:1712.02047*, 2017.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Kipf and Welling, 2017] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.

[Li *et al.*, 2016] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard S. Zemel. Gated graph sequence neural networks. In *ICLR*, 2016.

[Mikolov *et al.*, 2010] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *INTERSPEECH 2010*, pages 1045–1048, 2010.

[Mou *et al.*, 2016] Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. Natural language inference by tree-based convolution and heuristic matching. In *ACL*, pages 130–136, 2016.

[Palangi *et al.*, 2015] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using the long short term memory network: Analysis and application to information retrieval. *In Proceedings of the 31th International Conference on Machine learning*, 2015.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.

[Qi *et al.*, 2018] Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D Manning. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, 2018.

[Rocktäschel *et al.*, 2016] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. Reasoning about entailment with neural attention. In *ICLR*, 2016.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[Tai *et al.*, 2015] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *ACL*, pages 1556–1566, 2015.

[Veličković *et al.*, 2018] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[Wu *et al.*, 2018] Wei Wu, Houfeng Wang, Tianyu Liu, and Shuming Ma. Phrase-level self-attention networks for universal sentence encoding. In *EMNLP*, pages 3729–3738, 2018.

[Xu and Raginsky, 2017] Aolin Xu and Maxim Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. In *NeurIPS*, pages 2524–2533, 2017.

[Xu *et al.*, 2015] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 2048–2057, 2015.

[Yoon *et al.*, 2018] Deunsol Yoon, Dongbok Lee, and SangKeun Lee. Dynamic self-attention: Computing attention over words dynamically for sentence embedding. *arXiv preprint arXiv:1808.07383*, 2018.

[Zeiler, 2012] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.