Deep Learning in Causality Mining

Subjects: Computer Science, Artificial Intelligence
Contributor: Wajid Ali

Deep learning models for causality mining (CM) can enhance the performance of learning algorithms, improve the processing time, and increase the range of mining applications.

Keywords: machine learning (ML) ; causality mining (CM) ; Neural Network (NN) ; Deep Learning ; Cause-Effect relationship classification

1. Neural Networks and Deep Learning

In the era of information processing tasks, machine learning (ML) has been merged in many disciplines, including information mining, relations classification, image processing, video classifications, recommendation, and analysis of different social networks. Including all ML algorithms, Neural Network (NN) and DL are identified as representation learning [1] extensively used. NN computes a result/predication/output, which generally states forward propagation (FF). During FF, the NN receives inputs vector X and result in a prediction vector Y. More generally, NN is based on interconnected layers (input, hidden, and output layer). Each layer is linked via a so-called weight matrix (W) to the next layer. Further, each layer consists of different combinations of neurons/nodes, where each node gets a particular number of inputs and computes a prediction/output. Every node in the output layers makes weighted addition based on received values from the input neurons. Further, the weighted addition is passed to some nonlinear activation functions (Sigmoid, Tan Hyperbolic (Tanh), Rectified Linear Unit (ReLU), Leaky ReLU, and Softmax Activation Function) to compute outputs. **Figure 1** represents a simple NN with one input layer, three hidden layers (H_1, H_2, and H_3), one output layer, and four weight matrices (W_1, W_2, W_3, and W_4).

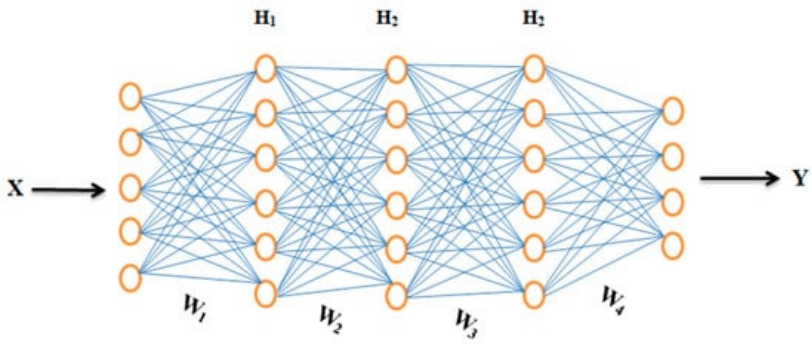

Figure 1. Represents a simple NN of one input layer, three hidden layers, and one output layer.

We set an input vector X to calculate dot-product by the first weight matrix (W_1) and used the nonlinear activation function to the result of this dot-product, which output a new vector h_1 that denotes values of the nodes in the first layer. Further, h_1 is used as a new input vector to the next layer, where similar operations are executed again. This process is repeated until the final output vector Y is produced, known as the NN prediction. While Equations (1)–(4) represent the whole set of operations in NN, where "σ" denotes an arbitrary activation function.

(1) $$\vec{h}_1 = \sigma\left(\vec{x}.\ W_1\right)$$

(2) $$\vec{h}_2 = \sigma\left(\vec{h}_1.\ W_2\right)$$

(3) $$\vec{h}_3 = \sigma\left(\vec{h}_1.\ W_3\right)$$

(4) $\overrightarrow{y} = \sigma\left(\overrightarrow{h}_3 . W_4\right)$

More generally, we consider NN as a function instead of using a combination of interconnected neurons. With this function, we combine all operations in a chained format in Equation (5) that we have seen in the above four equations.

(5) $\overrightarrow{y} = NN\left(\overrightarrow{x}\right) = \sigma\left(\sigma\left(\sigma\left(\overrightarrow{x} . W_1\right) . W_2\right) . W_3\right) . W_4\right)$

2. Loss Functions and Optimization Algorithms

The selection of loss function and optimization algorithms for DL networks can significantly generate optimal and quicker results. Every input in the feature vector is allocated its particular weight, which chooses the impact that the specific input desires in the summation function (Y). In simple words, certain inputs are made more significant than others by assigning them more weight, which has a superior effect in Y. Furthermore, a bias (b) is added to summation shown in Equation (6).

(6) $Y = bw_0 + w_1 x_1 + w_2 x_2 + w_3 x_3$

The outcome Y is a weighted sum is converted to performed output using a non-linear activation function (f_{NL}). In this case, the preferred result is the probability of an event, which is represented by Equation (7).

(7) $\hat{p} = f_{NL}\left(Y\right)$

In many learning models, error (e) is calculated as the gap between the actual and predicted results in Equation (8).

(8) $J\left(w\right) = p - \hat{p}$

The function for error calculation is called Loss Function J (.), which significantly affects the model prediction. Distinct J (.) will provide diverse errors for a similar prediction. Different J (.) deals with various problems, including classification, detection, extraction, and regression. Furthermore, error J (w) is a function of the network/model's inner parameters (weights and bias). Precise likelihoods require minimizing the calculated error. In NN, this is achieved by Back Propagation (BP) [2], in which the existing error is commonly propagated backward toward the preceding layer, where optimization function (OF), including Stochastic Gradient Decent (SGD), Adagrad, and Adam is used to modify the parameters in an efficient way to the minimize error.

The OF calculates the gradient (partial derivative) of J (.) concerning parameter (weights), and weights are improved in the reverse direction of the calculated gradient. This process is repetitive until it reaches the minimum J (.). Equation (9) represents the optimization process.

(9) $w^{(k+1)} = w^{(k)} = \frac{\partial}{\partial w^{(k)}} J\left(w\right)$

The basic differences between different models are based on the number of layers and the architecture of the interconnected nodes. In those models, neurons are structured into sequential layers, where each neuron receives inputs only from previous layers neurons, called Feedforward Neural Networks (FFNNs). Though, there is no clear consensus on precisely what explains a Deep Neural Network (DNN), networks with several hidden layers are known as deep and those with several layers are known as very deep [3]. Contrary to traditional and ML techniques, DL techniques have enhanced performance in computer vision (Image Processing, Video Processing, Audio Processing, and Speech Processing) [4][5][6], and NLP tasks (Text Classification, Information Retrieval, Event Prediction, Sentiment Analysis, and Language Translation) [7][8][9][10][11].

Usually, the effectiveness of shallow ML algorithms is based on the goodness of input data representation. Compared to precise data representation, the performance of depraved data representation is usually lower. Hence, for shallow ML tasks, feature engineering is an effective research direction in raw datasets and will lead to various research studies. Usually, most of the features are domain-dependent which need much human effort e.g., in computer vision tasks, diverse features are compared and proposed including Bag of Words (BoW), Scale Invariant Feature Transform (SIFT) [12], and Histogram of Oriented Gradients (HOG) [13]. Similarly, in NLP tasks, diverse features sets are used including BoW, Linguistics Patterns (LP), and Clue Terms (CT), Syntactic, and Semantic context. Contrary, DL techniques work on automatic feature engineering, which lets researchers get more discriminative features with minimal human effort and

domain knowledge [14]. As discussed above that DL techniques are based on a low-level, middle-level, and high-level layered structure for data representation, where the low-level layers are used for low-level features, the middle-level/hidden layers are used to extract hidden/middle-level features. Finally, the high-level features are extracted by high-level layers.

3. Motivation for Causality Mining

DL applications are resulted based on feature representation and algorithms together with the design. These are related to data illustration/representation and learning structure. For data illustration, there is typically a disjunction among what information is said to be essential for the task, against what illustration produces good outcomes. For instance, Syntactic Structure, Sentiment Analysis, Lexicon Semantics, and Context are supposed by some linguists to be of fundamental importance. However, prior works are based on bag-of-words (BoW) system proven satisfactory performance [15]. The BoW [16], frequently seen as vector space models, includes an illustration that accounts only for the words/tokens and their frequency of existence. BoW overlooks the order and relations of words and treats every token as a distinctive feature. BoW neglects syntactic format, still delivers effective results for what some could consider syntax-oriented applications. This judgment recommends that simple illustrations, when combined with a big data set, may work superior to difficult representations. These outcomes verify the argument courtesy of the significance of DL architectures and algorithms. Often the effective language modeling guarantees the advancement of NLP. The aim of statistical language designing is the probabilistic illustration of word sequences, which is a complex job because of the dimensionality curse. In [17], a breakthrough for language designing with NN aimed to overcome the dimensionality cures by learning a distributed illustration of tokens and giving a likelihood function for structures.

A significant challenge in NLP study, related to other areas including computer vision, looks complicated to reach an in-depth illustration of language using statistical/ML networks. A core task in NLP is to illustrate texts (documents), which comprises feature learning, i.e., mining expressive information to allow additional analysis and processing of raw data. Non-statistical approaches are based on handcrafted features engineering, which is time-consuming. Through, the development of algorithms needs careful human analysis to mine and exploit instances of such features. While, deep supervised approaches are more data-driven and can be used in extra general efforts, which directed a robust data illustration. In the presence of huge amounts of unlabeled datasets, unsupervised learning techniques are known to be critical tasks. With the beginning of DL and the sufficiency of unlabeled datasets, unsupervised techniques become a critical job for representation learning. At present, many NLP tasks depend on annotated data, while most unannotated data encourages study in employing deep data-driven unsupervised techniques. Given the possible power of DL techniques in NLP tasks, it looks critical to analyses numerous DL techniques extensively.

DL models have a hierarchical structure of layers that learn from data representation by input layer, then pass them through multiple intermediate layers (hidden layers) for further processing [18]. Finally, the last layer computes the output predation. ANN is a representative network using FP and backward propagation (BP). FP is used for processing weighted sum (WX) of input from the prior layer along with bias (b) term and further passes it to a sequence of Convolutional, Non-linear, Pooling, and Fully connected layers to produce the required output (final prediction). Equation (10) represents the fundamental matrices of the neural networks.

$$Z = A\left(WX + b\right) \tag{10}$$

where ‘W’ represents the weight (number matrix), also known as parameters, X represents the input feature vector, ‘b’ represents the bias term, ‘A’ represents the activation function, and Z represents the final prediction. Similarly, the BP computes the derivative/slope/gradient of an objective function by chain rule of the gradient to the weights of a multilayer stack of modules via the chain rule of derivatives. DL plays a role by deeply analyzing input and capturing all related features from low to high levels. The semantic configuration and representation learning are strengthened by neural processing and vector representation, making machines capable of feeding raw data to automatically determine hidden illustrations for final prediction [18] automatically. DL techniques have some fundamental strengths for CM, including, (1) By DL techniques, CM takes advantage of non-linear processing, which creates non-linear conversion from source to target output. They have the power to learn all related features from input data by a layered structure with different parameters and hyperparameters. (2) Compared to traditional and shallow ML techniques, DL can automatically capture important features without much human effort. (3) In the DL network, the optimization function plays an important role for the end-to-end paradigm to train a more complex task for CM. (4) With DL techniques, both data-driven and program-driven techniques are easily structured for CM tasks.

4. Deep Learning Frameworks

Currently, some well-known DL frameworks are available at hand for diverse model designing. Such frameworks are either the library or interface tools that help ML developers and research scientists to develop and design DL networks more efficiently. **Table 1** represent some well-known frameworks including Torch [19], TensorFlow [20], DeepLearning4j (DL4j) [21], Caffe [22], MXNet [23], Theano [24], Microsoft Cognitive Toolkit (CNTK) [25], Neon [26], Keras [27], and Gluon [28]. They all play a very significant role in DL architectures. Due to space limitations, it is advised for readers to visit [29] for detailed information about the mentioned Frameworks.

Table 1. Summary of Deep Learning Framework.

Frameworks	References	Primary Language	Interface Provision	RNN and CNN Provision	Key Note to Know About
Torch	[19]	**C and Lua**	**Python, C/C++, and Lua**	**Yes**	✓ Allow standard IDE for debugging, such as PyCharm or PDA ✓ It works with dynamically updated graph ✓ It is mostly used for DL applications, such as NLP and Computer Vision.
TensorFlow (TF)	[20]	**Python and C++**	**Python, Java, JavaScript, C/C++, Julia, C#, and Go**	**Yes**	✓ TF is the best choice for DL networks deployments ✓ Used for Data Integration (DI), such as SQL tables, input graphs, and images ✓ Along with deploying networks on influential computing clusters, TF can run networks on mobile systems (Android and iOS) as well.
DL4j	[21]	**Java, JVM**	**Python, Java, and Scala**	**Yes**	✓ It integrates the employment of the GloVe, Deep Autoencoder, Recursive Neural Tensor Network, Word2Vec, and Doc2Vec. ✓ It uses both Hadoop and Spark, this helps to accelerate network training. ✓ It trains neural networks in parallel through repeated reduction through clusters.
Caffe	[22]	**C++**	**MATLAB and Python**	**Yes**	✓ It is an open-source DL framework ✓ Works fine in computer vision ✓ It supports industrial and researchers applications

Frameworks	References	Primary Language	Interface Provision	RNN and CNN Provision	Key Note to Know About
MXNet	[23]	**//**	**Python, C++, Perl, R, Go, Matlab, Scala, and Julia.**	**Yes**	✓ It can support several GPUs with optimized calculation and fast context switching. ✓ It is a scalable and lean DL framework with the provision of previous networks including, CNNs, GRU, and LSTM. ✓ It supports symbolic and imperative programming.
Theano	[24]	**Python**	**Python**	**Yes**	✓ It lets to process mathematical operations such as multi-dimensional arrays ✓ It is used to handle computation for large algorithms used in DL ✓ It works well with GPU as compared to CPU
CNTK	[25]	**C++/C#**	**C++, Python, and BrainScript**	**Yes**	✓ It is an open-source app for commercial DL. ✓ It easily combines feed-forward deep neural networks, CNN, RNN, and LSTM. ✓ It describes the NN as a chain of computational stages through a directed graph
Neon	[26]	**Python**	**Python**	**Yes**	✓ It is an open-source DL framework ✓ It uses its own GPU and CPU backend ✓ It performs well on large batches
Keras	[27]	**Python**	**Python**	**Yes**	✓ User friendly, easy, and modular ✓ It offers the advantages of comprehensive adoption, provision for a wide range of incorporation with at least five back-end engines including, Theano, TensorFlow, PlaidML, CNTK, and MXNet ✓ Support several GPUs and distributed training

Frameworks	References	Primary Language	Interface Provision	RNN and CNN Provision	Key Note to Know About
Gluon	[28]	**Python**	**Python**	**Yes**	✓ Gluon provides a friendly API, for defining easy, clear, simple, and brief code ✓ It is easier for developers to understand and learn ✓ The model's definition is dynamic, it is easier to maintain because of its flexible structure.

5. Deep Learning Techniques for Causality Mining (CM)

Recently several works have been published, and most of the attention has been given to supervised systems such as shallow ML and DL approaches. The basic distinction among these systems is that advanced features engineering is essential for ML techniques, wherein DL techniques; features are learned automatically by training. However, previous approaches were largely automated, only focused on extracting explicit and simple implicit causality, and did not address complex implicit and ambiguous causalities. Furthermore, most of the early works have focused on identifying whether a relation or sentence is causal or not, and little attention is given to determine the direction of causality that which entity is the effect, and which one is the cause. The challenges mentioned above are critical for NLP researchers. Recently, DL techniques have been applied to various NLP tasks such as sentiment analysis, sentence classification, topic categorization [30], POS tagging, named entity recognition (NER), semantic role labeling (SRL), relation classification, and causality mining.

The two most widely used classifiers among various deep neural classifiers for relation classification are CNNs and RNN. In NLP, those classifiers are based on a discrete representation of words in vector space, known as word embedding that captures syntactic and semantic information of words [31][32]. The two most widely used classifiers among various deep neural classifiers for relation classification are CNNs and RNN. To the best of our knowledge, very few DL techniques are used for CM; some are discussed in this section. Similarly, **Figure 2** represents the processing levels of DL techniques, which consist of different phases of processing till to the final prediction. In this figure, the model is provided the raw input data, passed it to pre-processing steps for cleaning it for further processing. Further, the pre-processed data is passed to the input layer of the model and followed by multiple hidden layers for deep analysis of hidden features by using different hyperparameter settings. Finally, the output prediction is achieved at the output layer. If the prediction is correct, then the model is finalized. Otherwise, the model is trained repeatedly by applying the loss function to reduce the error until the final prediction is based on the model's performance evaluation metrics (precision, accuracy, and recall score).

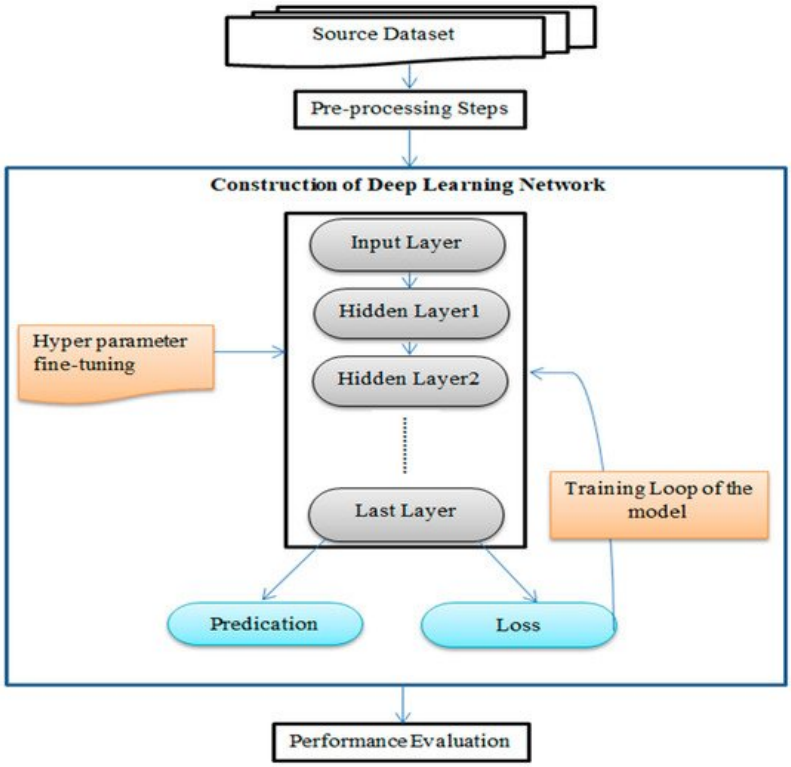

Figure 2. Processing level of DL techniques.

In [33], two networks are presented, a Knowledge-based features mining network and Deep CNN, to train a model for implicit and explicit causalities and their direction. They used sentence context for designing the problem into a three-class classification of entity pairs, including class-1 that specifies the annotated pair with causal direction e1 -> e2 (cause, effect), class-2 entity pairs with causal direction e2 -> e1 (effect, cause), and class-3 entity pairs are non-causal. A list of hypernyms in WordNet is prepared for each of the two annotated entities in a source sentence They used two labeled datasets including, SemEval-2007 Task-4 (http://docs.google.com/View?docID=w.df735kg3_8gt4b4c) and SemEval- 2010 Task-8" dataset (http://docs.google.com/View?docid=dfvxd49s_36c28v9pmw), in which total of 479 samples are used for class-1, 927 for class-2, and 982 for class-3. The SemEval-2007 dataset has seven labeled relations and the SemEval-2010 has nine relations, including cause-effect relation. They extract causality from each dataset as positive labeled data and extract a random mix of other relations as negative data.

Ref. [34] propose a novel technique using multi-column convolutional neural networks (MCNNs) and source background knowledge (BK) for CM. It is a variant of CNN [35] with several independent columns. The inspiration for this work was [36]. They used short binary patterns to connect pairs of nouns like "A causes B" and "A prevents B" to increase the performance of event causality recognition. They focused on such event causalities, "smoke cigarettes" → "die of lung cancer" by taking an original sentence from which the candidate of causalities is extracted with the addition of related BK taken from the web texts. Three distinct methods are used to get related texts for a given causality candidate from 4 billion web pages as a source of BK, including (1) Why-question answering, (2) Using Binary Pattern (BP), and (3) Clues Terms. These techniques identify useful BK scattered in the web archives and feed into MCNNs for CM. MCNNs consists of 8 columns, where five columns are used to process event causality candidates and their nearby contexts in the original sentence. The other three columns deal with web archives. Then the output of all columns based on their layers combination is combined into the last layer for final prediction. Using all types of BK (Base + BP + WH + CL), the top achieved average precision is 55.13%, which is 7.6% higher than the best of [36] methods (47.52%). Note that by extending single CNN's to multi-column CNN's (CNN-SENT vs. Base), the proposed work obtained a 5.6% improvement, and further gave 5.8% improvement by adding with external BK. [37] enhanced MCNN by adding causality attention (CA), which results in the CA-MCNN model. This model is based on two notions that enhanced why-QA, which includes expressing implicitly expressed causality in one text by explicit cues from other text and describing the causes of similar events by using a set of similar words.

In [38], a novel set of event semantics and position features are used to train a Feed-Forward Network (FFN) for implicit causality. This work aims to improve ANN with features that take assistance from linguistic and associated works. It captures knowledge about the position and content of events contained in the relation. They used Penn Discourse Treebank (PDTB) and CST News (CST-NC) corpus. The whole objective function of the proposed algorithm is shown in Equation (11).

$$J = -\sum_{n=1}^{\infty} \sigma \begin{pmatrix} W_2 || \max(\tanh(W_1 . (Xi . E \oplus Xe + b_1) \\ \oplus Xp) || + b_2) \log P(y) + \ell ||\theta|| 2 \end{pmatrix} \quad (11)$$

where the set of parameters is Θ= {E, W_1, b_1, W_2, b_2}, cross-entropy function is used for the loss function, which is regularized by the squared norm of parameters and scaled by hyperparameter (ℓ), positional features (Xp), input indices array (Xi), the true class label (y), and event-related features (Xe). **Table 1** lists the most popular DL approaches for CM based on their targets, architecture, datasets, and references. A neural encoder-decoder approach predicts causally related events in stories through standard evaluation framework choice of plausible alternatives (COPA) [39]. This was the first approach to evaluate a neural-based model for such kinds of tasks, which learns to predict relations between adjacent sequences in stories as a means of modeling causality.

The bi-LSTM [40] is a linguistically informed architecture for automatic CM using word linguistics features and word-level embedding. It contains three modules: linguistic preprocessor and feature extractor, resource creation, and prediction background for cause/effect. A causal graph is created after grouping and proper generalization of the extracted events and their relations. They used the BBC News Article dataset, a portion of SemEval2010 task-8 related to “Cause-Effect”, and adverse drug effect (ADE) dataset for training. In [41], a Temporal Causal Discovery Framework (TCDF), a DL model that learns temporal causal graph design by mining causality in continuous observational time series data. It applied multiple attention-based CNN along with a causal support step. It can also mine time interruption among cause and the existence of its effect. They used two benchmarks with multiple datasets including, simulated financial market and simulated functional magnetic resonance imaging (FMRI) data. Both contain a ground truth comprising the underlying causal graph. The experimental analysis shows that this mechanism is precise in mining time-series data.

Ref. [42] proposes a novel deep CNN using grammar tags for cause-effect pair identification from nominal words in natural language corpus knowledge reasoning. Though, the prior works mainly were based on predefined syntactic and linguistic rules. The modern approaches use shallow ML primarily Deep NN on top of linguistic and semantic knowledge to classify nominal word relations in a corpus. They used the SemEval-2010 Task 8 corpus for enhancing the performance of CM. In [43], a novel idea of Knowledge-Oriented CNN (K-CNN) for causality identification is presented. This model combined two channels: Data-Oriented Channel (DOC), which acquires important features of causality from the target data, and Knowledge-Oriented Channel (KOC), which integrates former human knowledge to capture the linguistic clues of causality. In KOC, the convolutional filters are automatically created from available knowledge bases (FrameNet and WordNet) without training the classifier by a huge amount of data. Such filters are the embedding of causation words. Additionally, it uses clustering, filters selection, and additional semantic features to increase the performance of K-CNN. They used three datasets including Causal-Time Bank4 (CTB), SemEval-2010 task-86, and Event StoryLine datasets7. More specifically, the KOC is used to integrate existing linguistic information from knowledge bases. Where DOC is used to learn important features from data by using a pre-defined convolutional filter. These two channels complement each other and extract valuable features of CM.

In the same year, a novel feed-forward neural network (FFNN) was used with a context word extension mechanism for CM in tweets [44]. For event context word extension, they used BK, extracted from news articles in the form of a causal network to identify event causality. They have used 2018 commonwealth game-related tweets held in Australia. This was a challenging job because tweets are mostly composed of unstructured nature, highly informal, and lack contextual information. This approach is closely related to [38] for detecting causality between events using FFNN by enhancing the feature set by computing distances among events trigger word and related words in the phrase. Though, such positional knowledge for tweets might not show the causal direction more easily because tweets are mostly composed of noisy words and characters e.g., # (hashtags), @ sign, question marks (?), URLs, and emojis. Hence, such data is not appropriate for the detection of causality in tweets. Inspired by [44], the automatic mining of causality in a short corpus is a useful and challenging task [45], because it contains many informal characters, emojis, and questions marks. This technique was applied a deep causal event detection and context word extension approach for CM in tweets. They used more than 207k tweets using Twitter API (https://developer.twitter.com/en/docs/tweets/search/overview). They prepare to collect those tweets that were associated with the “Commonwealth Games-2018 held in Australia”. This study [46] presents a BERT-based approach using multiple classifiers for CM inside a web corpus, which used independent labels given by multiple annotators in the corpus. By training multiple classifiers, hold all annotators procedure, where every classifier predicts the labels provided by a particular annotator, and integrate the result of all classifiers to predict the final labels found by the majority vote. BERT is a pre-trained network with a huge amount of corpus that learned some sort of BK for event-causal relations during pre-training. They used (Hashimoto et al., 2014) in the construction of source datasets. The experimentations prove that the performance is improved when BERT is pre-trained with a web corpus that covers a huge amount of event causalities instead of using Wikipedia texts. Though this effect was inadequate, hence, they further

enhanced the performance by simply adding corpus associated with an input causality candidate as a BK to the input of the BERTs, which significantly beat the state-of-the-art approach [34] by around 0.5 in average precision.

Ref. [47] explored the causality effect of search queries associated with bars and restaurants on every day new cases in the United State (US) areas with low and high everyday cases. GT searches for bars and restaurants presented a major effect on every day new cases for areas with higher numbers of every day new cases in the US. They used the deep LSTM model for training, which is a typical problem in ML tasks. In [48], the Event Causality identification (ECI) model are proposed by targeting the limitations of past approaches by leveraging outside knowledge for reasoning, which can significantly improve the illustration of events and also mine event-agnostic, context-specific patterns, by a mechanism named “event mention masking generalization”, which can significantly improve the capability of the model to handle new and previous unnoticed cases. Significantly, the important element of this model is “Knowledge-aware causal reasoned”, which can exploit BK in external CONCEPTNET knowledge bases [49] to improve the cognitive process. They used 3 benchmark datasets including, Causal-TimeBank, Event Story Line, and Event Causality for experimentations, which show the model achieves state-of-the-art performance. In [50], the problem of causal impact is considered for numerous ‘COVID-19’ associated policies on the outbreak dynamics in diverse US states at different time intervals in 2020. The core issue in this work is the presence of time-varying and overlooked confounders. To address this issue, they integrated data from several COVID-19 related databases comprising diverse types of information, which help as substitutions for confounders. They used a neural network-based approach, which learns the illustrations of the confounders using time-varying observational and relational data and then guesses the causal effect of such policies on the outbreak dynamics with the learned confounder representations. The outcomes of this study confirm the proficiency of the model in controlling confounders for causal valuation of COVID-19 associated policies.

In [51], a self-attentive Bi-LSTM-CRF based approach is presented, named Self-attentive BiLSTM-CRF wIth Transferred Embedding (SCITE). This technique formulates CM as a sequence tagging problem. This is useful for directly mining cause and effect events without considering cause-effect pairs and their relationship separately. Moreover, to progress the performance of CM, a multi-head self-attention procedure is presented into the model to acquire the dependencies among causal words. To solve two issues, first, they included Flair embedding due to prior information deficiency in the [52]. Second, in terms of positions in the text, cause and effect are rarely far from each other. For this, a multi-head self-attention [53] is applied. The SemEval 2010 task 8 is used with extended annotation, in which Flair-BiLSTM-CRF achieved progress of about 6.32% over the Bi-LSTM-CRF compared with BERT and ELMo (rises of 4.55% and 6.28%). Moreover, the causality tagging approach produced enhanced results compared to the general tagging approach under the SCITE model. This study [54] developed three network-architectures (Masked Event C-BERT, Event aware C-BERT, C-BERT) on the top of language models (pre-trained BERT) that influence the complete sentence context, events context, and events masked context for CM among expressed events in natural language text (NLT). They simply focus to recognize possible causality among marked events in a given sequence of text, but it doesn’t find the validity of such relations.

This approach achieved state-of-the-art performance in the proposed data distributions and can be used for mining causal diagrams and/or constructing a chain of events from an unstructured corpus. For experimentation, they generated their dataset from three benchmarks including, Semeval 2010 task 8 [55], Semeval 2007 task 4 [56], and ADE [57] corpus. This approach achieved state-of-the-art performance in the proposed data distributions and can be used for mining causal diagrams and/or constructing a chain of events from an unstructured corpus.

References

1. Deng, L. A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Trans. Signal Inf. Process. 2014, 3, 1–29.
2. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning Internal Representations by Error Propagation. In Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Foundations; MIT Press: Cambridge, CA, USA, 1986; pp. 399–421.
3. Schmidhuber, J. Deep learning in neural networks: An overview. Neural Netw. 2015, 61, 85–117.
4. LeCun, Y.; Neural, Y.B. Convolutional networks for images, speech, and time series. Handb. Brain Theory Neural Netw. 1995, 3361, 1995.
5. Burney, A.; Syed, T.Q. Crowd Video Classification Using Convolutional Neural Networks. In Proceedings of the 2016 International Conference on Frontiers of Information Technology (FIT), Islamabad, Pakistan, 19–21 December 2016; pp. 247–251.

6. Abdel-Hamid, O.; Mohamed, A.R.; Jiang, H.; Deng, L.; Penn, G.; Yu, D. Convolutional neural networks for speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. 2014, 22, 1533–1545.
7. Yan, Y.; Chen, M.; Saad Sadiq, M.; Shyu, M.-L. Efficient Imbalanced Multimedia Concept Retrieval by Deep Learning on Spark Clusters. Int. J. Multimed. Data Eng. Manag. IJMDEM 2017, 8, 1–20.
8. Yan, Y.; Chen, M.; Shyu, M. Deep learning for imbalanced multimedia data classification. In Proceedings of the 2015 IEEE international symposium on multimedia (ISM), Miami, FL, USA, 14–16 December 2015; pp. 483–488.
9. Kim, Y. Convolutional Neural Networks for Sentence Classification. arXiv 2016, arXiv:1408.5882.
10. Kalchbrenner, N.; Grefenstette, E.; Blunsom, P. A Convolutional Neural Network for Modelling Sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MA, USA, 22–27 June 2014; pp. 655–665.
11. Dos Santos, C.; Gatti, M. Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. In Proceedings of the COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, 23–29 August 2014; pp. 69–78.
12. Lowe, D.G. Object Recognition from Local Scale-Invariant Features. In Proceedings of the Eventh IEEE International Conference on Computer Vision, TKerkyra, Greece, 20–27 September 1999; pp. 1150–1157.
13. Dalal, N.; Histograms, B.T.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
14. Najafabadi, M.M.; Villanustre, F.; Khoshgoftaar, T.M.; Seliya, N.; Wald, R.; Muharemagic, E. Deep learning applications and challenges in big data analytics. J. Big Data 2015, 2, 1–21.
15. Pang, B.; Lee, L.; Vaithyanathan, S. Thumbs up? Sentiment Classification using Machine Learning Techniques. In Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002), Philadelphia, PA, USA, 6–9 July 2002; pp. 79–86.
16. Harris, Z.S. Distributional Structure. Word 1954, 10, 146–162.
17. Popov, M.; Kulnitskiy, B.; Perezhogin, I.; Mordkovich, V.; Ovsyannikov, D.; Perfilov, S.; Borisova, L.; Blank, V. A Neural Probabilistic Language Model. Fuller. Nanotub. Carbon Nanostruct. 2003, 3, 1137–1155.
18. Ian, G.; Bengio, Y.; Courville, A. Deep Learning; MIT Press: Cambridge, MA, USA, 2016; pp. 436–444.
19. Collobert, R.; Bengio, S.; Mariethoz, J. Torch: A Modular Machine Learning Software Library. Available online: http://publications.idiap.ch/downloads/reports/2002/rr02-46.pdf (accessed on 15 October 2020).
20. Abadi, M.; Agarwal, A.; Barham, E.B. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv 2016, arXiv:1603.04467.
21. Skymind Skymind. Deeplearning4j Deep Learning Framework. 2017. Available online: https://deeplearning4j.org/ (accessed on 16 October 2020).
22. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 22nd ACM international conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 675–678.
23. Chen, T.; Li, M.; Li, Y.; Lin, M.; Wang, N.; Wang, M.; Xiao, T.; Xu, B.; Zhang, C.; Zhang, Z. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. arXiv 2015, arXiv:1512.01274.
24. Al-Rfou, R. Theano: A Python Framework for Fast Computation of Mathematical Expressions. arXiv 2016, arXiv:1605.02688.
25. Agarwal, A.; Akchurin, E.; Basoglu, C.; Chen, G.; Cyphers, S.; Droppo, J.; Eversole, A.; Guenter, B.; Hillebrand, M.; Huang, X.; et al. An Introduction to Computational Networks and the Computational Network Toolkit. MSR-TR-2014-112 (DRAFT Vol.1.0). 2016, pp. 1–50. Available online: https://www.microsoft.com/en-us/research/wp-content/uploads/2014/08/CNTKBook-20160217.pdf (accessed on 1 October 2021).
26. NervanaSystems. The Neon Deep Learning Framework. Available online: https://github.com/NervanaSystems/neon (accessed on 11 May 2017).
27. Gulli, A.; Pal, S. Deep Learning with Keras; Packt Publishing Ltd.: Birmingham, UK, 2017.
28. Wood, M. Introducing Gluon: A New Library for Machine Learning from AWS and Microsoft: Introducing Gluon. 2017. Available online: https://aws.amazon.com/blogs/aws/introducing-gluon-a-new-library-for-machine-learning-from-aws-and-microsoft/ (accessed on 1 October 2021).

29. Pouyanfar, S.; Sadiq, S.; Yan, Y.; Tian, H.; Tao, Y.; Reyes, M.P.; Shyu, M.L.; Chen, S.C.; Iyengar, S.S. A survey on deep learning: Algorithms, techniques, and applications. ACM Comput. Surv. CSUR 2018, 51, 1–36.

30. Riaz, M.; Girju, R. Toward a Better Understanding of Causality between Verbal Events: Extraction and Analysis of the Causal Power of Verb-Verb Associations. In Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGdial), Metz, France, 22–24 August 2013; pp. 21–30.

31. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. arXiv 2013, arXiv:1301.3781.

32. Turian, J.; Ratinov, L.; Bengio, Y. Word representations: A simple and general method for semi-supervised learning. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, Uppsala, Sweden, 11–16 July 2010; pp. 384–394.

33. De Silva, T.N.; Zhibo, X.; Rui, Z.; Kezhi, M. Causal relation identification using convolutional neural networks and knowledge based features. Int. J. Comput. Syst. Eng. 2017, 11, 696–701.

34. Kruengkrai, C.; Torisawa, K.; Hashimoto, C.; Kloetzer, J.; Oh, J.-H.; Tanaka, M. Improving Event Causality Recognition with Multiple Background Knowledge Sources Using Multi-Column Convolutional Neural Networks. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017; pp. 3466–3473.

35. Ciresan, D.; Meier, U.; Schmidhuber, J. Multi-column Deep Neural Networks for Image Classification. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, Rhode Island, 16–21 June 2012; pp. 3642–3649.

36. Hashimoto, C.; Torisawa, K.; Kloetzer, J.; Sano, M. Toward future scenario generation: Extracting event causality exploiting semantic relation, coantext, and association features. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, MA, USA, 22–27 June 2014; pp. 987–997.

37. Oh, J.; Torisawa, K.; Kruengkrai, C.; Iida, R.; Kloetzer, J. Multi-column convolutional neural networks with causality-attention for why-question answering. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, Cambridge, CA, USA, 6–10 February 2017; pp. 415–424.

38. Ponti, E.M.; Korhonen, A. Event-related features in feedforward neural networks contribute to identifying causal relations in discourse. In Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics, Valencia, Spain, 3 April 2017; pp. 25–30.

39. Roemmele, M.; Gordon, A.S. An Encoder-decoder Approach to Predicting Causal Relations in Stories. In Proceedings of the First Workshop on Storytelling, New Orleans, Louisiana, 5 June 2018; pp. 50–59.

40. Dasgupta, T.; Saha, R.; Dey, L.; Naskar, A. Automatic extraction of causal relations from text using linguistically informed deep neural networks. In Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue, Melbourne, Australia, 12–14 July 2018; pp. 306–316.

41. Nauta, M.; Bucur, D.; Seifert, C. Causal Discovery with Attention-Based Convolutional Neural Networks. Mach. Learn. Knowl. Extr. 2019, 1, 312–340.

42. Ayyanar, R.; Koomullil, G.; Ramasangu, H. Causal Relation Classification using Convolutional Neural Networks and Grammar Tags. In Proceedings of the 2019 IEEE 16th India Council International Conference (INDICON), Marwadi University, Rajkot, India, 13–15 December 2019; pp. 1–3.

43. Li, P.; Mao, K. Knowledge-oriented Convolutional Neural Network for Causal Relation Extraction from Natural Language Texts. Expert Syst. Appl. 2019, 115, 512–523.

44. Kayesh, H.; Islam, M.S.; Wang, J. On Event Causality Detection in Tweets. arXiv 2019, arXiv:1901.03526.

45. Kayesh, H.; Islam, M.S.; Wang, J.; Kayes, A.S.M.; Watters, P.A. A deep learning model for mining and detecting causally related events in tweets. Concurr. Comput. Pract. Exp. 2020, e5938.

46. Kadowaki, K.; Iida, R.; Torisawa, K.; Oh, J.H.; Kloetzer, J. Event causality recognition exploiting multiple annotators' judgments and background knowledge. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 5816–5822.

47. Mehrabadi, M.A.; Dutt, N.; Rahmani, A.M. The Causality Inference of Public Interest in Restaurants and Bars on COVID-19 Daily Cases in the US: A Google Trends Analysis. JMIR Public Health Surveill. 2020, 7, 1–6.

48. Liu, J.; Chen, Y.; Zhao, J. Knowledge Enhanced Event Causality Identification with Mention Masking Generalizations. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), Yokohama, Japan, 7–15 July 2020; pp. 3608–3614.

49. Speer, R.; Lowry-Duda, J. ConceptNet at SemEval-2017 Task 2: Extending word embeddings with multilingual relational knowledge. arXiv 2017, arXiv:1704.03560.
50. Ma, J.; Dong, Y.; Huang, Z.; Mietchen, D.; Li, J. Assessing the Causal Impact of COVID-19 Related Policies on Outbreak Dynamics: A Case Study in the US. arXiv 2021, arXiv:2106.01315.
51. Li, Z.; Li, Q.; Zou, X.; Ren, J. Causality extraction based on self-attentive BiLSTM-CRF with transferred embeddings. Neurocomputing 2021, 423, 207–219.
52. Akbik, A.; Blythe, D.; Vollgraf, R. Contextual String Embeddings for Sequence Labeling. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 1638–1649.
53. Vaswani, A.; Brain, G.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention Is All You Need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
54. Khetan, V.; Ramnani, R.; Anand, M.; Sengupta, S.; Fano, A.E. Causal-BERT: Language models for causality detection between events expressed in text. arXiv 2021, arXiv:2012.05453v2, 965–980.
55. Hendrickx, I.; Kim, S.N.; Kozareva, Z.; Nakov, P.; Diarmuidó, D.; Diarmuidó, S.; Padó, S.; Pennacchiotti, M.; Romano, L.; Szpakowicz, S. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations Between Pairs of Nominals. arXiv 2019, arXiv:1911.10422.
56. Girju, R.; Nakov, P.; Nastase, V.; Szpakowicz, S.; Turney, P.; Yuret, D. Semeval-2007 task 04: Classification of semantic relations between nominals. In Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007), Prague, Czech Republic, 23 June 2007; pp. 13–18.
57. Gurulingappa, H.; Rajput, A.; Roberts, A.; Fluck, J. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. J. Biomed. 2012, 45, 885–892.
