Natural Language Engineering (2019), **25**, pp. 519–541
doi:10.1017/S1351324919000238

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Neural embeddings: accurate and readable inferences based on semantic kernels

Danilo Croce*, Daniele Rossini and Roberto Basili

Department of Enterprise Engineering, University of Roma, Tor Vergata, Rome, Italy
*Corresponding author. Email: croce@info.uniroma2.it

Abstract

Sentence embeddings are the suitable input vectors for the neural learning of a number of inferences about content and meaning. Similarity estimation, classification, emotional characterization of sentences as well as pragmatic tasks, such as question answering or dialogue, have largely demonstrated the effectiveness of vector embeddings to model semantics. Unfortunately, most of the above decisions are epistemologically opaque as for the limited interpretability of the acquired neural models based on the involved embeddings. We think that any effective approach to meaning representation should be at least epistemologically coherent. In this paper, we concentrate on the *readability* of neural models, as a core property of any embedding technique consistent and effective in representing sentence meaning. In this perspective, this paper discusses a novel embedding technique (the Nyström methodology) that corresponds to the reconstruction of a sentence in a kernel space, inspired by rich semantic similarity metrics (a semantic kernel) rather than by a language model. In addition to being based on a kernel that captures grammatical and lexical semantic information, the proposed embedding can be used as the input vector of an effective neural learning architecture, called *Kernel-based deep architectures* (*KDA*). Finally, it also characterizes *by design* the KDA explanatory capability, as the proposed embedding is derived from examples that are both human readable and labeled. This property is obtained by the integration of KDAs with an explanation methodology, called *layer-wise relevance propagation (LRP)*, already proposed in image processing. The Nyström embeddings support here the automatic compilation of argumentations in favor or against a KDA inference, in form of an explanation: each decision can in fact be linked through LRP back to the real examples, that is, the landmarks linguistically related to the input instance. The KDA network output is explained via the analogy with the activated landmarks. Quantitative evaluation of the explanations shows that richer explanations based on semantic and syntagmatic structures characterize convincing arguments, as they effectively help the user in assessing whether or not to trust the machine decisions in different tasks, for example, Question Classification or Semantic Role Labeling. This confirms the epistemological benefit that Nyström embeddings may bring, as linguistically rich and meaningful representations for a variety of inference tasks.

Keywords: readable inference; semantic kernels; neural embeddings of sentences

1. Introduction

Nonlinear methods such as Deep Neural Networks achieve state-of-the-art performances in several semantic NLP tasks (Collobert *et al*. 2011; Goldberg 2016). The wide spread of Deep Learning is supported by the impressive results and their feature learning capability (Bengio, Courville, and Vincent 2013; Kim 2014): input words and sentences are usually modeled as dense embeddings (i.e., vectors or tensors), whose dimensions correspond to latent semantic concepts acquired during an unsupervised pretraining stage. In similarity estimation, classification, emotional characterization of sentences as well as pragmatic tasks, such as question answering or dialogue, they largely demonstrated their effectiveness to model semantics.

Unfortunately, several drawbacks arise. First, most of the above approaches are epistemologically opaque as for the limited interpretability of the acquired neural models based on the involved embeddings. Second, injecting linguistic information into an NN without degrading its transparency properties is still a problem with much room for improvement. Word embeddings are widely adopted as an effective pretraining approach, although there is no general agreement about how to provide deeper linguistic information to the NN. Some structured NN models have been proposed (Hochreiter and Schmidhuber 1997; Socher *et al.* 2013), although usually tailored to specific problems. Recursive NNs (Socher *et al.* 2013) have been shown to learn dense feature representations of the nodes in a structure, thus exploiting similarities between nodes and sub-trees. Also, long-short term memory (LSTM) networks (Hochreiter and Schmidhuber 1997) build intermediate representations of sequences, resulting in similarity estimates over sequences and their inner subsequences. In general, such intermediate representations are strongly task dependent: this is beneficial from an engineering standpoint, but certainly controversial from a linguistic and cognitive point of view. In recent years, many approaches proposed extensions to the previous methods. Semi-supervised models within the multitask learning paradigm have been investigated (Collobert *et al.* 2011). Context-aware dense representations (Pennington, Socher, and Manning 2014) and deep representations based on sub-words or characters (Devlin *et al.* 2018; Peters *et al.* 2018) successfully model syntactic and semantic information. Linguistically informed mechanisms have been proposed to train the self-attention to attend syntactic information in a sentence, granting state-of-the-art results in Semantic Role Labeling (Strubell *et al.* 2018). However, in such approaches, the captured linguistic properties are never made explicit and the complexity of learned latent spaces only exacerbates the interpretability problem. Hence, despite state-of-the-art performances, such approaches are not a solution for a straightforward understanding of the linguistic aspects that are responsible for a network decisions. Attempts to solve the interpretability problem of NNs have been proposed in computer vision (Erhan, Courville, and Bengio 2010; Bach *et al.* 2015), but their extension to the NLP scenario is not straightforward.

We think that any effective approach to meaning representation should be at least epistemologically coherent, that is, readable and justified through an argument theoretic lens on interpretation. This means that inferences based on vector embeddings should also naturally correspond to a clear and uncontroversial logical counterpart: in particular, neurally trained semantic inferences should be also epistemologically transparent. In other words, neural embeddings should support model readability, that is, to trace back *causal connections* between the implicitly expressed linguistic properties of an input instance and the classification output produced by a model. Meaning representation should thus strictly support the (neural) learning of epistemologically well-founded models.

In this paper, we concentrate on this *readability* issue, as a core property of any meaning representation. In this view, we propose to provide *explicit information* regarding semantics by relying on linguistic properties of sentences, that is, by modeling the lexical, syntactic, and semantic constraints implicitly encoded in the linguistic structure. A natural choice, which we will adopt in this paper, is represented by learning methods based on tree kernels (TKs; Collins and Duffy 2001; Shawe-Taylor and Cristianini 2004; Moschitti 2012) as the feature space they capture reflects linguistic patterns. Approximation method can then be used to successfully map tree structures into dense vector representations useful to train a neural network. As suggested in Croce *et al.* (2017), the Nyström dimensionality reduction method (Williams and Seeger 2001) is of particular interest as it allows to reconstruct a low-dimensional embeddings of the rich kernel space by computing kernel similarities between input examples and a set of selected instances, called *landmarks*. If methods such as Nyström's are used over TKs, the projection vectors will encode information captured by such kernels, which have been proved to incorporate syntactic as well as semantic materials (Croce, Moschitti, and Basili 2011). Kernels play the role of inner products in complex (i.e., highly, and possibly infinitely, dimensional) spaces. They suggest linguistically principled metrics. Although they do not provide directly vector or tensor-like representations, they can be

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

used to model semantics and train effective algorithms. Moreover, embeddings are a solution to map them into useful vector representations. Linguistic structures (e.g., parse trees) expressed by kernels can be used in the training of an NN, that is in form of vectors or tensors, as suggested by Croce *et al.* (2017). The resulting vectors can be then used as input of an effective neural learner, namely a *Kernel-based deep architecture* (KDA).

KDA has been shown beneficial by Croce *et al.* (2017) as the Nyström- based low-rank embedding of input sentences has been used as the early layer of a deep feed-forward network, achieving state-of-the-art results in different semantic tasks, such as Question Classification and Semantic Role Labeling. While the Nyström-based methodology corresponds to the reconstruction of a sentence in a kernel space, it must be stressed that it expresses a rich linguistically justified metrics (through the underlying semantic kernel) rather than a language model, as most embedding method tend to do (e.g., Mikolov *et al.* 2013). Moreover, the proposed embedding corresponds to a linear combination of a set of randomly chosen independent instances (i.e., the *landmarks*), as they are represented in the kernel space. This property also characterizes *by design* the KDA ability to explain its decisions: it is obtained by integrating the neural decision carried out with a model of the activation state of a network, called *layer-wise relevance propagation* (*LRP*): this is a method, proposed in image processing, to explain a neural decision, as the detection of the state of activation of some components of the network, that is, the contribution of input layers (and nodes) to the fired output. We can apply the same process to the KDA decision and detect which components of a Nyström embedding (i.e., the landmarks) are mostly activated. A KDA can automatically compile argumentations in favor or against its inference: each decision is in fact linked back to the real examples, through LRP, and these are the landmarks most linguistically related to the input instance. The KDA network output is thus explained via the *analogy with the activated landmarks*. Quantitative evaluation of these explanations shows that richer explanations based on semantic and syntagmatic structures characterize convincing arguments in different tasks, for example, Question Classification or Semantic Role Labeling, in the sense that they provide right assistance to the user in accepting or rejecting the system output. This confirms the epistemological benefit that Nyström embeddings may bring, as linguistically rich and meaningful representations for a variety of inference tasks.

In this paper, we first survey approaches to improve the transparency of neural models in Section 2. We present the role of linguistic similarity principles as they are expressed by Semantic Kernels in Section 3.2. The Nyström methodology and the KDA architecture are defined in Section 3.3 and Section 4.1, respectively, while the explanation model based on the KDA architecture is defined in Section 4.3. A method for the quantitative evaluation of explanations is defined in Section 4.4. The evaluation over two tasks is discussed in Sections 5.1 and 5.2, for performance in semantic inference and explanation quality, respectively. Finally, Section 6 summarizes achievements, open issues, and future directions of this work.

2. Related work on interpretability

Advancements in Deep Learning are allowing the penetration of data-driven models into areas that have profound impacts on society, as health care services, criminal justice systems, and financial markets. Consequently, the traditional criticism of epistemological opaqueness of AI-based systems has recently drawn much attention from the research community, as the ability for humans to understand models and suitable weight the assistance they provide is a central issue for the correct adoption of such systems. However, to empower neural models with interpretability property is still an open problem as it even lacks a broad consensus on the definitions of interpretability and explanation.

In Lipton (2018), Chakraborty *et al.* (2017) analyzed definitions of interpretability and transparency found in literature and structured them across two main dimensions: *Model*

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

Transparency, that is, understanding the mechanism by which the model works, and *Post-Hoc Explanability* (or Model Functionality), that is, the property by which the system convey to its users information useful to justify its functioning such as intuitive evidence supporting the output decisions. The latter can be further divided into *global* explanations, that is, a description of the full mapping the network has learned, and *local* explanations, that is, motivations underlying a single output. Examples of global explanations are methods that use deconvolutional networks to characterize high-layer units in a CNN for image classification (Zeiler and Fergus 2013) and approaches that derive an identity for each filter in a CNN for text classification, in terms of the captured semantic classes (Jacovi, Sar Shalom, and Goldberg 2018).

Some Local Post-Hoc Explanation methods provide visual insights, for example, through a GAN-generated image to assess the information detail of deep representations extracted from the input text (Spinks and Moens 2018); however, as these methods stemmed from efforts into making neural image classifiers more *readable*, they are usually designed to trace back the portions of the network input that mostly contributed to the output decision. Network propagation techniques are used to identify the patterns of a given input item (e.g., an image) that are linked to the particular deep neural network prediction as in Erhan, Courville, and Bengio (2010) and Zeiler and Fergus (2013). Usually, these are based on backward algorithms that layer-wise reuse arc weights to propagate the prediction from the output down to the input, thus leading to the recreation of *meaningful* patterns in the input space. Typical examples are deconvolution heatmaps, used to approximate through Taylor series the partial derivatives at each layer (Simonyan, Vedaldi, and Zisserman 2013), or the so-called LRP that redistributes back positive and negative evidence across the layers (Bach *et al.* 2015).

Several efforts have been made in the perspective of providing explanations of a neural classifier, often by focusing into highlighting an handful of crucial features (Baehrens *et al.* 2010) or deriving simpler, more readable models from a complex one, for example, a binary decision tree (Frosst and Hinton 2017), or by local approximation with linear models (Ribeiro *et al.* 2016). However, although they can explicitly show the representations learned in the specific hidden neurons (Frosst and Hinton 2017), these approaches base their effectiveness on the user ability to establish the quality of the reasoning and the accountability, as a side effect of the quality of the selected features: this can be very hard in tasks where no strong theory about the decision is available or the boundaries between classes are not well defined. Sometimes, explanations are associated with vector representations as in Ribeiro *et al.* (2016), that is, bag-of-word in case of text classification, which is clearly weak at capturing significant linguistic abstractions, such as the involved syntactic relations. When embeddings are used to trigger neural learning the readability of the model is a clear proof of the consistency of the adopted vectors as meaning representations, as clear understanding of what a numerical representation is describing allows human inspectors to assess whether the machine correctly modeled the target phenomena or not. Readability here refers to the property of a neural network to support *linguistically motivated explanations* about its (textual) inference. A recent methodology exploits the coupling of the classifier with some sort of generator, or decoder, responsible for the selection of output justifications: Lei, Barzilay, and Jaakkola (2016) propose a generator that provides rationales for a multi-aspect sentiment analysis prediction by highlighting short and self-sufficient phrases in the original text.

Concerns in the research area of deriving interpretable, sparse representations from dense embeddings (Faruqui *et al.* 2015; Subramanian *et al.* 2018) have recently grown: for example, in Trifonov *et al.* (2018) an effective unsupervised approach to disentangle meanings from embedding dimensions as well as automatic evaluation method have been proposed. In this work, we present a model generating *local post-hoc explanations* through analogies with previous real examples by exploiting the LRP extended to a linguistically motivated neural architecture, the KDA, that exhibits a promising level of epistemological transparency. With respect to the works above, our proposal holds a few nice properties. First, the instance representations corresponds to the

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

similarity scores modeled by the semantic tree kernels with real examples, i.e. the landmarks. These exemplify the general linguistic properties (e.g. trees and lexical embeddings) and the task-relevant information (i.e., the target class): this allows [...] neural discriminator. Second, it is well suited to deal with short texts, where it may be difficult to highlight meaningful, yet not trivial, portions of input as justifications, as well as with the classification of segments of longer text (e.g., multi-aspect sentiment analysis) in a fashion similar to the one described for SRL in Section 5.1.2. Moreover, it provides explanations that are easily interpretable even by nonexpert users, as they are inspired and expressed at language level: these are done by entire sentences and allow the human inspector to implicitly detect lexical, semantic, and syntactic connections in the comparison, and consequently judge the trustworthiness of the decision, relying only on his/her linguistic competence. Lastly, the explanation-generation process is computationally inexpensive, as the LRP corresponds to a single pass of backward propagation. As discussed in Section 4.3, it provides a transparent and epistemologically coherent view on the system's decision.

3. Kernel-based learning in semantic inferences

3.1 Kernels as nonlinear feature mappings

Prediction techniques such as support vector machines (SVMs) learn decision surfaces that correspond to hyper-planes in the original feature space by computing inner products between input examples; consequently, they are inherently linear and cannot discover nonlinear patterns in data. A possible solution is to use a mapping $\phi : x \in \mathbb{R}^n \mapsto \phi(x) \in F \subseteq \mathbb{R}^N$ such that nonlinear relations in the original space become linearly separable in the target projection space, enabling the SVM to correctly separate the data by computing inner products $\langle \phi(x_i), \phi(x_j) \rangle$ in the new feature space. However, such projections can be computationally intense. *Kernel functions* are a class of functions that allow to compute $\langle \phi(x_i), \phi(x_j) \rangle$ without explicitly accessing the input representation in the projection space. Formally, given a feature space X and a mapping ϕ from X to F, a kernel κ is any function satisfying

$$\kappa(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle \quad \forall x_i, x_j \in X \tag{1}$$

An important generalization result is the Mercer Theorem (Shawe-Taylor and Cristianini 2004), stating that for any symmetric positive semi-definite function κ there exists a mapping ϕ such that 1 is satisfied. Hence, kernels include a broad class of functions (Shawe-Taylor and Cristianini 2004). Research community has been exploring kernel methods for decades and a wide variety of kernel paradigms have been proposed. In the following subsections, we will illustrate advancements in TKs, as they are well suited to encode formalisms, such as dependency graphs or grammatical trees, traditionally exploited in the linguistics communities.

3.2 Semantic kernels

Learning to solve NLP tasks usually involves the acquisition of decision models based on complex semantic and syntactic phenomena. For instance, in Paraphrase Detection, verifying whether two sentences are valid paraphrases involves rewriting rules in which the syntax plays a fundamental role. In Question Answering, the syntactic information is crucial, as largely demonstrated in Croce, Moschitti, and Basili (2011). Similar needs are applicable to the Semantic Role Labeling task that consists in the automatic discovery of linguistic predicates (together with their corresponding arguments) in texts. A natural approach to such problems is to apply Kernel methods (Robert Müller, Mika, Rätsch, Tsuda, and Schölkopf 2001; Shawe-Taylor and Cristianini 2004) that have been traditionally proposed to decouple similarity metrics and learning algorithms in order to alleviate the impact of feature engineering in inductive processes. Kernels may directly

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

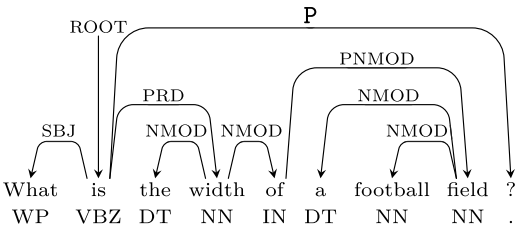

Figure 1. Dependency parse tree of "*What is the width of a football field?*".

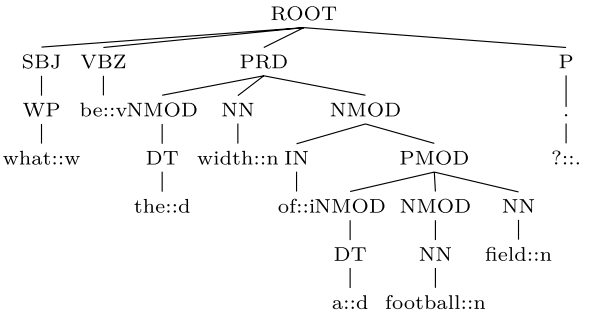

Figure 2. Grammatical Relation Centered Tree (GRCT) of *"What is the width of a football field?"*.

operate on complex structures and then be used in combination with linear learning algorithms, such as SVMs (Vapnik 1998). Sequence (Cancedda, Gaussier, Goutte, and Renders 2003) or TKs (Collins and Duffy 2001) are of particular interest as the feature space they capture reflects linguistic patterns. A sentence s can be represented as a parse tree that expresses the grammatical relations implied by s: parse trees are extracted by using the Stanford Parser (Manning, Surdeanu, Bauer, Finkel, Bethard, and Mc-Closky 2014). TKs (Collins and Duffy 2001) can be employed to directly operate on such parse trees, evaluating the tree fragments shared by the input trees. This operation corresponds to a dot product in the implicit feature space of all possible tree fragments. Whenever the dot product is available in the implicit feature space, kernel-based learning algorithms, such as SVMs (Cortes and Vapnik 1995), can operate in order to automatically generate robust prediction models. TKs thus allow estimating the similarity among texts, directly from sentence syntactic structures, that can be represented by parse trees. The underlying idea is that the similarity between two trees T_1 and T_2 can be derived from the number of shared tree fragments. Let the set $\mathcal{T} = \{t_1, t_2, \ldots, t_{|\mathcal{T}|}\}$ be the space of all the possible substructures and $\chi_i(n_2)$ be an indicator function that is equal to 1 if the target t_i is rooted at the node n_2 and 0 otherwise. A TK function over T_1 and T_2 is defined as follows: $TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$ where N_{T_1} and N_{T_2} are the sets of nodes of T_1 and T_2, respectively, and $\Delta(n_1, n_2) = \sum_{k=1}^{|\mathcal{T}|} \chi_k(n_1)\chi_k(n_2)$ which computes the number of common fragments between trees rooted at nodes n_1 and n_2. The feature space generated by the structural kernels obviously depends on the input structures. Note that different tree representations embody different linguistic theories and may produce more or less effective syntactic/semantic feature spaces for a given task.

Many available linguistic resources are enriched with formalisms dictated by Dependency grammars and produce a significantly different representation as exemplified in Figure 1. Since TKs are not tailored to model the labeled edges that are typical of dependency graphs, these latter are rewritten into explicit hierarchical representations. Different rewriting strategies are possible, as discussed in Croce, Moschitti, and Basili (2011): a representation that is shown to be effective in several tasks is the grammatical relation centered tree (GRCT) illustrated in Figure 2: the PoS-Tags are children of grammatical function nodes and direct ancestors of their associated lexical items. Another possible representation is the Lexical only centered tree (LOCT) shown in Figure 3, which contains only lexical nodes and the edges reflect some dependency relations.

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

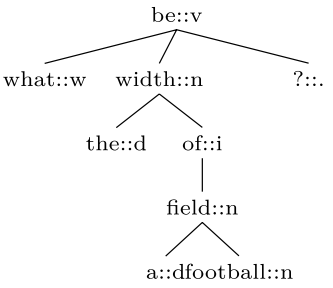

Figure 3. Lexical only centered tree (LOCT) of "*What is the width of a football field?*".

Different TKs can be defined according to the types of tree fragments considered in the evaluation of the matching structures. Subset of trees are exploited by the *subset tree kernel* (Collins and Duffy 2001), which is usually referred to as syntactic tree kernel (STK); they are more general structures since their leaves can be also nonterminal symbols. The subset trees satisfy the constraint that grammatical rules cannot be broken and every tree exhaustively represents a CFG rule. *Partial tree kernel* (PTK; Moschitti 2006) relaxes this constraint considering partial trees, that is, fragments generated by the application of partial production rules (e.g., sequences of nonterminal nodes with gaps). The strict constraint imposed by the STK may be problematic especially when the training dataset is small and only few syntactic tree configurations can be observed. Overcoming this limitation, the PTK usually leads to higher accuracy, as shown by Moschitti (2006).

Capitalizing lexical semantic information in convolution kernels. The TKs introduced in previous section perform a hard match between nodes when comparing two substructures. In NLP tasks, when nodes are words, this strict requirement reflects in a too strict lexical constraint that poorly reflects semantic phenomena, such as the synonymy of different words or the polysemy of a lexical entry. To overcome this limitation, we adopt Distributional models of Lexical Semantics (Schütze 1993; Sahlgren 2006; Padó and Lapata 2007) to generalize the meaning of individual words by replacing them with geometrical representations (also called Word Embeddings) that are automatically derived from the analysis of large-scale corpora (Mikolov *et al.* 2013). These representations are based on the idea that words occurring in the same contexts tend to have similar meaning: the adopted distributional models generate vectors that are similar when the associated words exhibit a similar usage in large-scale document collections. Under this perspective, the distance between vectors reflects semantic relations between the represented words, such as paradigmatic relations, for example, quasi-synonymy.[a] These word spaces allow to define meaningful soft matching between lexical nodes, in terms of the distance between their representative vectors. As a result, it is possible to obtain more informative kernel functions, which are able to capture syntactic and semantic phenomena through grammatical and lexical constraints. Moreover, the supervised setting of a learning algorithm (such as SVM), operating over the resulting kernel, is augmented with the word representations generated by the unsupervised distributional methods, thus characterizing a cost-effective semi-supervised paradigm.

The *smoothed partial tree kernel* (SPTK) described in Croce *et al.* (2011) exploits this idea extending the PTK formulation with a similarity function σ between nodes:

$$\Delta_{SPTK}(n_1, n_2) = \mu\lambda\sigma(n_1, n_2)\,, \quad \text{if } n_1 \text{ and } n_2 \text{ are leaves}$$

$$\Delta_{SPTK}(n_1, n_2) = \mu\sigma(n_1, n_2)\left(\lambda^2 + \sum_{\vec{I}_1, \vec{I}_2 : l(\vec{I}_1) = l(\vec{I}_2)} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)} \prod_{k=1}^{l(\vec{I}_1)} \Delta_{SPTK}\left(c_{n_1}\left(i_k^1\right), c_{n_2}\left(i_k^2\right)\right)\right) \tag{2}$$

[a] In such spaces, vectors representing the nouns *football* and *soccer* will be near (as they are synonyms according to one of their senses), while *football* and *dog* are far.

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

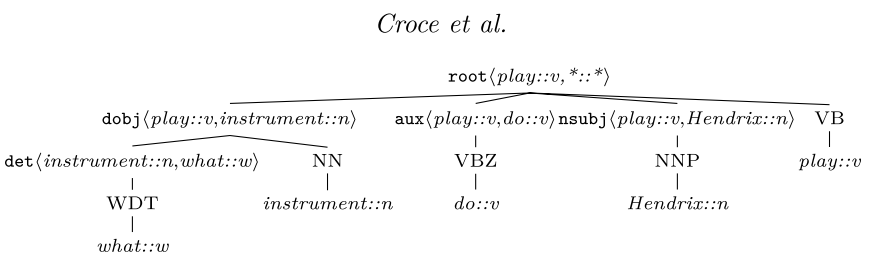

Figure 4. Compositional grammatical relation centered tree (CGRCT) of the sentence "*What instrument does Hendrix play?*".

In the SPTK formulation, the similarity function $\sigma(n_1, n_2)$ between two nodes n_1 and n_2 can be defined as follows:

- if n_1 and n_2 are both lexical nodes, then $\sigma(n_1, n_2) = \sigma_{LEX}(n_1, n_2) = \tau \frac{\vec{v}_{n_1} \cdot \vec{v}_{n_2}}{\|\vec{v}_{n_1}\| \|\vec{v}_{n_2}\|}$. It is the cosine similarity between the word vectors $\vec{v}_{n_1}$ and $\vec{v}_{n_2}$ associated with the labels of n_1 and n_2, respectively. τ is called *terminal factor* and weighs the contribution of the lexical similarity to the overall kernel computation.
- else if n_1 and n_2 are nodes sharing the same label, then $\sigma(n_1, n_2) = 1$.
- else $\sigma(n_1, n_2) = 0$.

The decay factors λ and μ are responsible for penalizing large child subsequences (that can include gaps) and partial sub-trees that are deeper in the structure, respectively.

Dealing with compositionality in TKs. The main limitations of the SPTK are that (*i*) lexical semantic information only relies on the vector metrics applied to the leaves in a context-free fashion and (*ii*) the semantic compositions between words are neglected in the kernel computation that only depends on their grammatical labels.

In Annesi, Croce, and Basili (2014), a solution for overcoming these issues is proposed. The pursued idea is that the semantics of a specific word depends on its context. For example, in the sentence, "What instrument does Hendrix play?", the role of the word *instrument* is fully captured if its composition with the verb *play* is taken into account. Such combination of lexical semantic information can be directly expressed into the tree structures, as shown in Figure 4. The resulting representation is a compositional extension of a GRCT structure, where the original label d_n of grammatical function nodes n (i.e., dependency relations in the tree) is augmented by also denoting their corresponding head/modifier pairs (h_n, m_n).

In CGRCTs, a (sub)tree rooted at dependency nodes can be used to provide a contribution to the kernel that is a function of the composition of vectors, $\vec{h}$ and $\vec{m}$, expressing the lexical semantics of the head h and modifier m, respectively. Several algebraic functions have been proposed in Annesi *et al.* (2014) to compose the vectors of h=l^h::pos^h and m=l^m::pos^m into a vector $\vec{c}^{h,m}$ representing the head modifier pair $c = \langle$l^h::pos^h,l^m::pos$^m\rangle$, in line with the research on Compositional Distributional Semantics (e.g., Mitchell and Lapata 2010). In this work, we investigated the additive function (according to the notation proposed in Mitchell and Lapata 2010) that assigns to a head/modifier pair c the vector resulting from the linear combination of the vectors representing the head and the modifier, that is, $\vec{c}^{h,m} = \alpha\vec{h} + \beta\vec{m}$. Although this composition method is very simple and efficient, it actually produces very effective kernel functions, as demonstrated in Annesi *et al.* (2014) and Filice *et al.* (2015). According to the CGRCT structures, Annesi *et al.* (2014) define the compositionally smoothed partial tree kernel (CSPTK). The core novelty of the CSPTK is the compositionally enriched estimation of the function σ. The function σ can be applied to lexical nodes, to POS tag nodes as well as to augmented dependency nodes. In the algorithm the

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

three cases are defined. For simple lexical nodes, σ consists of a lexical kernel σ_{LEX}, such as the cosine similarity between word vectors (sharing the same POS-tag): this is equivalent to Croce *et al.* (2011). For POS nodes σ consists of the identity function that is 1 only when the same POS is matched and it is 0 elsewhere.

The novelty of CSPTK corresponds to the compositional treatment of two dependency nodes, $n_1 = \langle d_1, h_1, m_1 \rangle$ and $n_2 = \langle d_2, h_2, m_2 \rangle$. The similarity function σ in this case corresponds to a compositional function σ_{Comp} between the two nodes. σ_{Comp} is not null only when the two nodes exhibit the same dependency relation, that is, $d = d_1 = d_2$, so that also the respective heads and modifiers share the same POS labels: this allows to exploit, case by case, the suitable contextual meaning of polysemous words, e.g. *bank*. In all these cases a compositional metric is applied over the two involved (h_i, m_i) compounds. In the simple case, the cosine similarity between the two vectors $\vec{c}_i^{\,h_i,m_i} = \alpha\vec{h}_i + \beta\vec{m}_i$, i=1,2, is applied. Other metrics correspond to more complex compositions $\Psi((\vec{h}_1, \vec{m}_1), (\vec{h}_2, \vec{m}_2))$ that account for linear algebra operators among the four vectors.

3.3 Approximating kernel spaces through the Nyström method

Given an input training dataset $\mathcal{D}$ of objects $o_i, i = 1 \ldots N$, a kernel $K(o_i, o_j)$ is a similarity function over $\mathcal{D}^2$ that corresponds to a dot product in the implicit kernel space, that is, $K(o_i, o_j) = \Phi(o_i) \cdot \Phi(o_j)$. The advantage of kernels is that the projection function $\Phi(o) = \vec{x} \in \mathbb{R}^n$ is never explicitly computed (Shawe-Taylor and Cristianini 2004). In fact, this operation may be prohibitive when the dimensionality n of the underlying kernel space is extremely large, as for TKs (Collins and Duffy 2001). Kernel functions are used by learning algorithms, such as SVM, to operate only implicitly on instances in the kernel space, by never accessing their explicit definition. Let us apply the projection function Φ over all examples o_i from $\mathcal{D}$ to derive representations, $\vec{x}_i$ denoting the ith row of the matrix $\mathbf{X}$. The Gram matrix can always be computed as $\mathbf{G} = \mathbf{X}\mathbf{X}^\top$, with each single element corresponding to $\mathbf{G}_{ij} = \Phi(o_i)\Phi(o_j) = K(o_i, o_j)$. The aim of the Nyström method (Drineas and Mahoney 2005) is to derive a new low-dimensional embedding $\tilde{\vec{x}}$ in a l-dimensional space, with $l \ll n$ so that $\tilde{\mathbf{G}} = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top$ and $\tilde{\mathbf{G}} \approx \mathbf{G}$. This is obtained by generating an approximation $\tilde{\mathbf{G}}$ of $\mathbf{G}$ using a subset of l columns of the Gram matrix, that is, the kernel evaluations between all the objects $\in \mathcal{D}$ and a selection of a subset $L \subset \mathcal{D}$ of the available examples, called *landmarks*. Suppose we randomly sample l columns of $\mathbf{G}$, and let $\mathbf{C} \in \mathbb{R}^{N \times l}$ be the matrix of these sampled columns. Then, we can rearrange the columns and rows of $\mathbf{G}$ and define $\mathbf{X} = [\mathbf{X}_1 \;\; \mathbf{X}_2]$ such that:

$$\mathbf{G} = \mathbf{X}\mathbf{X}^\top = \begin{bmatrix} \mathbf{W} & \mathbf{X}_1^\top\mathbf{X}_2 \\ \mathbf{X}_2^\top\mathbf{X}_1 & \mathbf{X}_2^\top\mathbf{X}_2 \end{bmatrix} \quad \text{and} \quad \mathbf{C} = \begin{bmatrix} \mathbf{W} \\ \mathbf{X}_2^\top\mathbf{X}_1 \end{bmatrix}$$

where $\mathbf{W} = \mathbf{X}_1^\top\mathbf{X}_1$, that is, the subset of $\mathbf{G}$ that contains only landmarks and $\mathbf{C}$ kernel evaluations between landmarks and the remaining examples. The Nyström approximation can be defined as

$$\mathbf{G} \approx \tilde{\mathbf{G}} = \mathbf{C}\mathbf{W}^\dagger\mathbf{C}^\top \tag{3}$$

where $\mathbf{W}^\dagger$ denotes the Moore–Penrose inverse of $\mathbf{W}$. The singular value decomposition (SVD) is used to obtain $\mathbf{W}^\dagger$ as follows. First, $\mathbf{W}$ is decomposed so that $\mathbf{W} = \mathbf{U}\mathbf{S}\mathbf{V}^\top$, where $\mathbf{U}$ and $\mathbf{V}$ are both orthogonal matrices, and $\mathbf{S}$ is a diagonal matrix containing the (nonzero) singular values of $\mathbf{W}$ on its diagonal. Since $\mathbf{W}$ is symmetric and positive definite, it holds that $\mathbf{W} = \mathbf{U}\mathbf{S}\mathbf{U}^\top$. Then, $\mathbf{W}^\dagger = \mathbf{U}\mathbf{S}^{-1}\mathbf{U}^\top = \mathbf{U}\mathbf{S}^{-\frac{1}{2}}\mathbf{S}^{-\frac{1}{2}}\mathbf{U}^\top$ and Equation (3) can be rewritten as

$$\mathbf{G} \approx \tilde{\mathbf{G}} = \mathbf{C}\mathbf{U}\mathbf{S}^{-\frac{1}{2}}\mathbf{S}^{-\frac{1}{2}}\mathbf{U}^\top\mathbf{C}^\top = (\mathbf{C}\mathbf{U}\mathbf{S}^{-\frac{1}{2}})(\mathbf{C}\mathbf{U}\mathbf{S}^{-\frac{1}{2}})^\top = \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\top \tag{4}$$

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

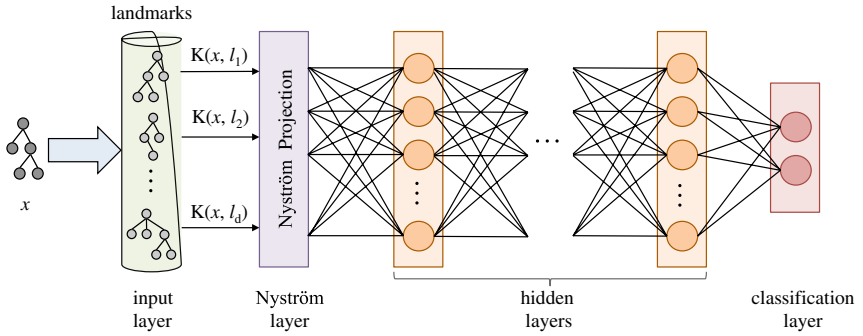

Figure 5. Kernel-based deep architecture.

which explicitates the desired approximation of G in terms of the described decomposition. Given an input example $o \in \mathcal{D}$, a new low-dimensional representation $\tilde{\vec{x}}$ can be thus determined by considering the corresponding item of $\mathbf{C}$ as

$$\tilde{\vec{x}} = \vec{c}\,\mathbf{U}\mathbf{S}^{-\frac{1}{2}} \tag{5}$$

where $\vec{c}$ is the vector whose jth individual component contains the evaluation of the kernel function between o and the landmark $o_j \in L$. Therefore, the method produces l-dimensional vectors.

4. Explainable neural learners through kernel embeddings

4.1 Kernel-based deep architectures

As discussed in Section 3.3, the Nyström representation $\tilde{\vec{x}}$ of any input example o is linear and can be adopted to feed a neural network architecture. We assume a labeled dataset $\mathcal{L} = \{(o, y) \mid o \in \mathcal{D},\ y \in Y\}$ being available, where o refers to a generic instance and y is its associated class. In this section, we define a multilayer perceptron (MLP) architecture, with a specific Nyström layer based on the Nyström embeddings of Equation (5). We will refer to this architecture, shown in Figure 5, as KDA. KDA has an *input layer*, a *Nyström layer*, a possibly empty sequence of nonlinear *hidden layers* and a final *classification layer*, which produces the output.

The *input* layer corresponds to the input vector $\vec{c}$, that is, the row of the $\mathbf{C}$ matrix associated with an example o. Note that, for adopting the KDA, the values of the matrix $\mathbf{C}$ should be all available. In the training stage, these values are in general cached. During the classification stage, the $\vec{c}$ vector corresponding to an example o is directly computed by l kernel computations between o and each of the l landmarks.

The input layer is mapped to the *Nyström* layer, through the projection in Equation (5). Note that the embedding provides also the proper weights, defined by $\mathbf{U}\mathbf{S}^{-\frac{1}{2}}$, so that the mapping can be expressed through the Nyström matrix $\mathbf{H}_{Ny} = \mathbf{U}\mathbf{S}^{-\frac{1}{2}}$: it corresponds to a pretrained stage derived through SVD, as discussed in Section 3.3. Equation (5) provides a static definition for $\mathbf{H}_{Ny}$ whose weights can be left invariant during the neural network training. However, the values of $\mathbf{H}_{Ny}$ can be made available for the standard back-propagation adjustments applied for training. Formally, the low-dimensional embedding of an input example, o, is $\tilde{\vec{x}} = \vec{c}\,\mathbf{H}_{Ny} = \vec{c}\,\mathbf{U}\mathbf{S}^{-\frac{1}{2}}$.

The resulting outcome $\tilde{\vec{x}}$ is the input to one or more nonlinear *hidden* layers. Each tth hidden layer is realized through a matrix $\mathbf{H}_t \in \mathbb{R}^{h_{t-1} \times h_t}$ and a bias vector $\vec{b}_t \in \mathbb{R}^{1 \times h_t}$, whereas h_t denotes the desired hidden-layer dimensionality. Clearly, given that $\mathbf{H}_{Ny} \in \mathbb{R}^{l \times l}$, $h_0 = l$. The first hidden

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

layer in fact receives in input $\vec{\tilde{x}} = \vec{c}\,\mathbf{H}_{Ny}$, which corresponds to $t = 0$ layer input $\vec{x}_0 = \vec{\tilde{x}}$ and its computation is formally expressed by $\vec{x}_1 = f(\vec{x}_0\mathbf{H}_1 + \vec{b}_1)$, where f is a nonlinear activation function, here a Rectified Linear Unit (ReLU). In general, the generic tth layer is modeled as

$$\vec{x}_t = f(\vec{x}_{t-1}\mathbf{H}_t + \vec{b}_t) \tag{6}$$

The final layer of KDA is the *classification layer*, realized through the output matrix $\mathbf{H}_O$ and the output bias vector $\vec{b}_O$. Their dimensionality depends on the dimensionality of the last hidden layer (called O_{-1}) and the number $|Y|$ of different classes, that is, $\mathbf{H}_O \in \mathbb{R}^{h_{O_{-1}} \times |Y|}$ and $\vec{b}_O \in \mathbb{R}^{1 \times |Y|}$, respectively. In particular, this layer computes a linear classification function with a softmax operator so that $\hat{y} = softmax(\vec{x}_{O_{-1}}\mathbf{H}_O + \vec{b}_O)$.

In order to avoid overfitting, two different regularization schemes are applied. First, the dropout is applied to the input $\vec{x}_t$ of each hidden layer ($t \geq 1$) and to the input $\vec{x}_{O_{-1}}$ of the final classifier. Second, a L_2 regularization is applied to the norm of each layer.

Finally, the KDA is trained by optimizing a loss function made of the sum of two factors: first, the cross-entropy function between the gold classes and the predicted ones; second the L_2 regularization, whose importance is regulated by a meta-parameter λ. The final loss function is thus

$$L(y, \hat{y}) = \sum_{(o,y)\in\mathcal{L}} y\, log(\hat{y}) + \lambda \sum_{\mathbf{H}\in\{\mathbf{H}_t\}\cup\{\mathbf{H}_O\}} ||\mathbf{H}||^2$$

where $\hat{y}$ are the softmax values computed by the network and y are the true one-hot encoding values associated with the example from the labeled training dataset $\mathcal{L}$.

4.2 Layer-wise relevance propagation

LRP (presented in Bach *et al.* 2015) is a framework which allows to decompose the prediction of a deep neural network computed over a sample, for example, an image, down to relevance scores for the single input dimensions of the sample such as subpixels of an image.

More formally, let $f : \mathbb{R}^d \to \mathbb{R}^+$ be a positive real-valued function taking a vector $x \in \mathbb{R}^d$ as input. The function f can quantify, for example, the probability of x being in a certain class. The LRP assigns to each dimension, or feature, x_d a relevance score $R_d^{(1)}$ such that:

$$f(x) \approx \sum_d R_d^{(1)} \tag{7}$$

Features whose score is $R_d^{(1)} > 0$ or $R_d^{(1)} < 0$ correspond to evidence in favor or against, respectively, the output classification. In other words, LRP allows to identify fragments of the input playing key roles in the decision, by propagating relevance backwards. Let us suppose to know the relevance score $R_j^{(l+1)}$ of a neuron j at network layer $l+1$, then it can be decomposed into messages $R_{i\leftarrow j}^{(l,l+1)}$ sent to neurons i in layer l:

$$R_j^{(l+1)} = \sum_{i\in(l)} R_{i\leftarrow j}^{(l,l+1)} \tag{8}$$

Hence, it derives that the relevance of a neuron i at layer l can be defined as

$$R_i^{(l)} = \sum_{j\in(l+1)} R_{i\leftarrow j}^{(l,l+1)} \tag{9}$$

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

Note that 8 and 9 are such that 7 holds. In this work, we adopted the ϵ-rule defined in Bach *et al.* (2015) to compute the messages $R_{i\leftarrow j}^{(l,l+1)}$:

$$R_{i\leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j + \epsilon \cdot \text{sign}(z_j)} R_j^{(l+1)}$$

where $z_{ij} = x_i w_{ij}$ and $\epsilon > 0$ is a numerical stabilizing term and must be small. The informative value is justified by the fact that the weights w_{ij} are linked to the weighted activations of the input neurons.

If we apply it to a KDA processing linguistic observations, then LRP implicitly traces back the syntactic, semantic, and lexical relations between the example and the landmarks; thus, it selects the landmarks whose presences were the most influential to identify the predicted structure in the sentence. Indeed, each landmark is uniquely associated with an entry of the input vector $\vec{c}$, as illustrated in Section 4.1.

4.3 KDA embeddings and model readability

Justifications for the KDA decisions can be obtained by explaining the evidence in favor or against a class using landmarks $\{\ell\}$ as examples. The idea is to select those $\{\ell\}$ that the LRP method produces as the most active elements in layer 0 during the classification. Once such active landmarks are detected, an *Explanatory Model* is the function in charge to compile a linguistically fluent explanation by using analogies or differences with the input case. The semantic expressiveness of such analogies makes the resulting explanation clear and increases the user confidence on the system reliability. When a sentence s is classified, LRP assigns activation scores r_ℓ^s to each individual landmark ℓ: let $\mathcal{L}^{(+)}$ (or $\mathcal{L}^{(-)}$) denote the set of landmarks with positive (or negative) activation score.

Formally, every explanation is characterized by a triple $e = \langle s, C, \tau \rangle$ where s is the input sentence, C is the predicted label, and τ is the modality of the explanation: $\tau = +1$ for positive (i.e., acceptance) statements, while $\tau = -1$ corresponds to rejections of the decision C. A landmark ℓ is *positively activated* for a given sentence s if there are not more than $k-1$ other active landmarks ℓ' whose activation value is higher than the one for ℓ, that is,

$$|\{\ell' \in \mathcal{L}^{(+)} : \ell' \neq \ell \wedge r_{\ell'}^s \geq r_\ell^s > 0\}| < k$$

Similarly, a landmark ℓ is *negatively activated* when:

$$|\{\ell' \in \mathcal{L}^{(-)} : \ell' \neq \ell \wedge r_{\ell'}^s \leq r_\ell^s < 0\}| < k$$

where k is a parameter used to make explanation depending on not more than k landmarks, denoted by $\mathcal{L}_k$. Positively (or negative) active landmarks in $\mathcal{L}_k$ are assigned to an activation value $a(\ell, s) = +1\ (-1)$. $a(\ell, s) = 0$ for all other not activated landmarks.

Given the explanation $e = \langle s, C, \tau \rangle$, a landmark ℓ whose (known) class is C_ℓ is *consistent* (or *inconsistent*) with e according to the fact that the following function:

$$\delta(C_\ell, C) \cdot a(\ell, q) \cdot \tau$$

is positive (or negative, respectively), where $\delta(C', C) = 2\delta_{kron}(C' = C) - 1$ and δ_{kron} is the Kronecker delta. The *explanatory model* is then a function $M(e, \mathcal{L}_k)$ which maps an explanation e, a subset $\mathcal{L}_k$ of the active *and* consistent landmarks $\mathcal{L}$ for e into a sentence f in natural language. Note that the value of k determines the amount of consistent landmarks and hence it regulates the trade-off between the capacity of the system to produce an explanation at all and the adherence of such explanation to the machine inference process: low values of k grant that the Model generates explanations using landmarks with high activation scores only; however, they may also result in the Model being unable to produce any explanation for some decisions, that is, when no consistent landmark is available.

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

Of course several definitions for $M(e, \mathcal{L}_k)$ are possible. A general explanatory model would be

$$M(e, \mathcal{L}_k) = M(\langle s, C, \tau \rangle, \mathcal{L}_k) = \begin{cases} \text{“}s \text{ is } C \text{ since it recalls me of } \ell\text{”} \\ \forall \ell \in \mathcal{L}_k^+ \quad \text{if } \tau > 0 \\ \\ \text{“}s \text{ is not } C \text{ since it does not recall me of} \\ \ell \text{ which is } C\text{”} \\ \forall \ell \in \mathcal{L}_k^- \quad \text{if } \tau < 0 \\ \\ \text{“}s \text{ is } C \text{ but I don't know why”} \\ \text{if } \mathcal{L} \equiv \emptyset \end{cases}$$

where $\mathcal{L}_k^+$ and $\mathcal{L}_k^-$ are the partition of landmarks with positive and negative relevance scores in $\mathcal{L}_k$, respectively.
Here we defined three explanatory models we used during experimental evaluation:

(Basic Model) The first model is the simplest. It returns an analogy only with the (unique) consistent landmark with the highest positive score if $\tau = 1$ and lowest negative when $\tau = -1$. In case no active and consistent landmark can be found, the Basic model returns a phrase stating only the predicted class, with no explanation. For example, given the triple $e_1 = \langle$'Put this plate in the center of the table', THEME$_{\text{PLACING}}$, $1\rangle$, that is an explanation for the Argument Classification task, the model would produce the following sentence:

> *I think* **“this plate”** *is* THEME *of* PLACING *in “Robot* PUT ***this plate*** *in the center of the table” since it reminds me of* **“the soap”** *in “Can you* PUT ***the soap*** *in the washing machine?”.*

(Multiplicative Model) In a second model, denoted as *multiplicative*, the system makes reference to up to $k_1 \leq k$ analogies with positively active and consistent landmarks. Given the above explanation e_1, and $k_1 = 2$, it would return:

> *I think* **“this plate”** *is* THEME *of* PLACING *in “Robot* PUT ***this plate*** *in the center of the table” since it reminds me of* **“the soap”** *in “Can you* PUT ***“the soap”*** *in the washing machine?” and it also reminds me of* **“my coat”** *in “*HANG ***my coat*** *in the closet in the bedroom”.*

(Contrastive Model) The last proposed model is more complex since it returns both a positive analogy (whether $\tau = 1$) and a negative ($\tau = -1$) analogy by selecting, respectively, the most positively relevant and the most negatively relevant consistent landmark. For instance, it could return:

> *I think* **“this plate”** *is the* THEME *of* PLACING *in “Robot* PUT ***this plate*** *in the center of the table” since it reminds me of* **“the soap”** *which is in “Can you* PUT ***the soap*** *in the washing machine” and it is not the* GOAL *of* PLACING *since different from* **“on the counter”** *in “*PUT *the plate* ***on the counter****”.*

All three models find their foundations, from an argumentation theory perspective, in the argument by analogy schema (Walton, Reed, and Macagno 2008): as such a kind of arguments gains strength proportionally to the linguistic plausibility of the analogy, a user exposed to it will implicitly gauge the evidences from the linguistic properties shared between the input sentence (or its parts) and the one used for comparison as well their importance with respect to the output decision, hence endowing a different amount of trust in the machine verdict accordingly.

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

4.4 Using information theory for validating explanations

In general, judging the semantic coherence of an explanation is a very difficult task. In this section, we propose an approach which aims at evaluating the quality of the explanations in terms of the amount of information that a user would gather given an explanation with respect to a scenario where such explanation is not made available. More formally, let $P(C|s)$ and $P(C|s, e)$ be, respectively, the prior probability of the user believing that the classification of s is correct and the probability of the user believing that the classification of s is correct given an explanation. Note that both indicate the level of confidence the user has in the classifier (i.e., the KDA) given the amount of available information, that is, with and without explanation. Three kinds of explanations are possible:

- **Useful explanations:** these are explanations such that C is correct and $P(C|s, e) > P(C|s)$ or C is not correct and $P(C|s, e) < P(C|s)$
- **Useless explanations:** they are explanations such that $P(C|s, e) = P(C|s)$
- **Misleading explanations:** they are explanations such that C is correct and $P(C|s, e) < P(C|s)$ or C is not correct and $P(C|s, e) > P(C|s)$

The core idea is that semantically coherent and exhaustive explanations must indicate correct classifications, whereas incoherent or nonexistent explanations must hint toward wrong classifications. Given the above probabilities, we can measure the quality of an explanation by computing the *Information Gain* (Kononenko and Bratko 1991) achieved: the *posterior* probability is expected to grow w.r.t. to the *prior* one for correct decisions when a good explanation is available against the input sentence while decreasing for bad or confusing explanations. The intuition behind Information Gain is that it measures the amount of information (provided in number of bits) gained by the explanation about the decision of accepting the system decision about an incoming sentence s. A positive gain indicates that the probability amplifies toward the right decisions and declines with errors. We will let users to judge the quality of the explanation and assign them a posterior probability that increases along with better judgments. In this way, we have a measure of how convincing is the system about its decisions as well as how weak is the system to clarify erroneous cases. To compare the overall performance of the different explanatory models M, the Information Gain is measured against a collection of explanations generated by M and then normalized throughout the collection's entropy E as follows:

$$I_r = \frac{1}{E} \frac{1}{|\mathcal{T}_s|} \sum_{j=1}^{|\mathcal{T}_s|} I(j) = \frac{I_a}{E} \tag{10}$$

where $\mathcal{T}_s$ is the explanations collection and $I(j)$ is the Information Gain of explanation j.

5. Experimental investigations

To assess the effectiveness of our approach on both discriminative power and interpetability improvement, we focused on two common tasks in semantic inferences: Question Classification (QC) and the Argument Classification (AC) step in the Semantic Role Labeling chain. Whereas performances in semantic inferences have been evaluated by the classic metrics, that is, accuracy, we devised the qualitative evaluation of generated explanations as a human manual task, which will be well described in Section 5.2. In fact, even if some automatic evaluation approaches have been proposed, as in Trifonov *et al.* (2018), the interpretability measurement problem is still controversial and no consensus on machine-executable methodology has been reached.

As details on performances will be illustrated in the following section, here we would like to stress that the proposed approach is *fully scalable*: (i) the computational intensive SVD has

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

reduced cost as it needs to be performed over the l landmarks only, resulting in $\mathcal{O}(l^2)$ with $l \ll n$, whereas the cost of a single Nystrom projection is $\mathcal{O}(kl + l^2)$, which can be reduced to $\mathcal{O}(kl)$ with k being the number of operations for a single kernel computation. (ii) Both the network computations and the operations for reconstructing the projection vector $\vec{c}$ can be parallelized. (iii) The computation of relevance attributes of input dimension has a cost comparable to a single forward pass throughout the network.

5.1 Training the KDA for complex semantic inferences

We conducted an extensive experimental investigation in order to demonstrate that the proposed KDA is an effective solution for combining the expressiveness of kernel methods with the powerful learning capabilities of Deep Learning. Furthermore, we will show that the KDA is very efficient and that it can easily scale to large datasets. Finally, we investigated the impact of linguistic information on the performance reachable by a KDA by studying the benefits that different kernels (each characterized by a growing expressive power) can bring to the accuracy in semantic inference tasks. We adopted the same architecture, without major differences, for both tasks, that is, QC and AC, and the good performances obtained in these rather different tasks clearly confirm that the proposed framework is a general solution with an extremely large applicability.

General experimental settings: the Nyström projector has been implemented in the KeLP framework (Filice *et al*. 2018). The neural network has been implemented in Tensorflow,[b] with two hidden layers whose dimensionality corresponds to the number of involved Nyström landmarks. The *ReLU* is the nonlinear activation function in each layer. The dropout has been applied in each hidden layer and in the final classification layer. The values of the dropout parameter and the λ parameter of the L_2-regularization have been selected from a set of values via grid-search. The Adam optimizer with a learning rate of 0.001 has been applied to minimize the loss function, with a multi-epoch (500) training, each fed with batches of size 256. We adopted an early stop strategy, where the best model was selected according to the performance over the development set. Every performance measure is obtained against a specific sampling of the Nyström landmarks with fixed sizes. Results averaged against 5 such samplings are always hereafter reported. In the following experiments, the only difference in the KDA configuration is the adopted kernels that will be described specifically for each task.

5.1.1 Semantic inferences: Question classification

QC is the task of mapping a question into a closed set of answer types in a Question Answering system. The adopted UIUC dataset (Li and Roth 2006) includes a training and test set of 5,452 and 500 questions, respectively, organized in six classes (like `ENTITY` or `HUMAN`). TKs resulted very effective, as shown in Croce, Moschitti, and Basili (2011) and Annesi, Croce, and Basili (2014).

A first experiment aims at understanding the impact of different kernels into the proposed KDA framework. The input vectors for the KDA are modeled using the Nyström method (with different kernels) based on a number of landmarks ranging from 100 to 1000. We tried different kernels with increasing expressiveness:

- BOWK: a liner kernel applied over bag-of-words vectors having lemmas as dimensions. It provides a pure lexical similarity.
- PTK: the partial tree kernel over the GRCT representations. It provides a lexical and syntactic similarity.
- SPTK: the smoothed partial tree kernel over the GRCT representations. It improves the reasoning of the PTK by including the semantic information derived by word embeddings.

[b] https://www.tensorflow.org/

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

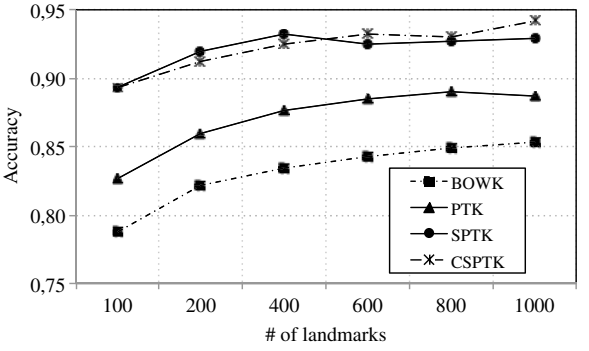

Figure 6. QC task—accuracy measure curves w.r.t. the number of landmarks.

- CSPTK: the compositionally smoothed partial tree kernel over the GRCT representations. It adds the semantic compositionality to the SPTK.

In the SPTK and CSPTK, we used 250-dimensional word vectors generated by applying the Word2vec tool with a Skip-gram model (Mikolov *et al.* 2013) to the entire Wikipedia. The TKs have default parameters (i.e., $\mu = \lambda = 0.4$).

Figure 6 shows the impact of different kernels in the proposed KDA model, whereas the different plots are based to a varying number of landmarks in the Nyström formulation. The increasing complexity of the investigated kernels directly reflects on the accuracy achieved by the KDA. The BOWK is the simplest kernel and obtains poor results: it needs 800 landmarks to reach 85% of accuracy.

The contribution of the syntactic information provided by TKs is straightforward. The PTK achieves about 90% of accuracy starting from 600 landmarks. These results are significantly improved by SPTK and CSPTK when the semantic information of the word embeddings is employed: even when only 100 landmarks are used, the KDA using these kernels can obtain 90% of accuracy and overcomes 94% with more landmarks. These achievements demonstrate that the KDA results directly depend on the involved kernel functions and that the improvement guaranteed by using a more expressive kernel cannot be obtained by the nonlinear learning of the Neural Network.

We also performed a second set of experiments to show that (*i*) the proposed KDA is far more efficient than a pure kernel-based approach, and (*ii*) the powerful nonlinear learning provided by the neural networks is necessary to take the best from the Nyström embeddings and achieve higher accuracies. In this case, we focused on the most accurate kernel, that is, the CSPTK. The kernel-based SVM formulation by Chang and Lin (2011), fed with the CSPTK (hereafter SVM_{ker}), is here adopted to determine the reachable upper bound in classification quality, that is, a 95% of accuracy, at higher computational costs. It establishes the state-of-the-art over the UIUC dataset. The resulting model includes 3,873 support vectors: this corresponds to the number of kernel operations required to classify any input test question.

To justify the need of the Neural Network, we compared the proposed KDA to an efficient linear SVM that is directly trained over the Nyström embeddings. This SVM implements the Dual Coordinate Descent method (Hsieh *et al.* 2008) and will be referred as SVM_{lin}.

Results are reported in Table 1: computational saving refers to the percentage of avoided kernel computations with respect to the application of the SVM_{ker} to classify each test instance. We also measured the performance of the Convolutional Neural Network[c] (CNN) of Kim (2014) that

[c]The deep architecture presented in Kim (2014) outperforms several NN models, including the Recursive Neural Tensor Network or Tree-LSTM presented in Socher *et al.* (2013) and Tai, Socher, and Manning (2015) which presents a semantic compositionality model that exploits parse trees. A higher result is shown in Zhang, Lee, and Radev (2016) where a CNN is combined with a Recursive Neural Networks in the so-called DSCNN, leading to an accuracy of 95.4%: unfortunately, this last work is not evaluated using the official train/test split and a direct comparison is not easily feasible.

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

Table 1. Results in terms of accuracy and saving in the QC task. In brackets, accuracy scores of the linear classifier SVM_{lin}

Model	#Land.	Accuracy	Saving
CNN (Kim 2014)	–	93.6%	–
LSTM (Zhou *et al*. 2015)	–	93.2%	–
BiLSTM (Zhou *et al*. 2015)	–	93.0%	–
C-LSTM (Zhou *et al*. 2015)	–	94.6%	–
SVM_{ker}	–	95.0%	0.0%
	100	88.5% (84.1%)	97.4%
	200	92.2% (88.7%)	94.8%
	400	93.7% (91.6%)	89.7%
KDA (SVM_{lin})	600	**94.3%**(92.8%)	**84.5%**
	800	94.3% (93.0%)	79.3%
	1,000	94.2% (93.6%)	74.2%

Notes: Accuracy scores of the linear classifier SVM_{lin} are given in brackets.

achieves the remarkable accuracy of 93.6%. Zhou *et al.* (2015) report results from a different version of LSTM, including a Bidirectional LSTM (BiLSTM) and the combination of a Convolutional and a Recurrent Neural Network (namely C-LSTM) that leads to an Accuracy of 94.6%.

Note that the linear classifier SVM_{lin} operating over the approximated kernel space achieves the same classification quality of the CNN when only 1000 landmarks are considered. KDA improves these results, achieving 94.3% accuracy even with fewer landmarks (only 600), showing the effectiveness of nonlinear learning over the Nyström input. Although SVM_{ker} improves to 95%, KDA provides a saving of more than 84% kernel computations at classification time. This result is straightforward as it confirms that (1) *linguistic information encoded in a tree is important in the analysis of questions*, (2) Nyström vectors correspond to *very expressive sentence embeddings,* and (3) they can be used effectively in the *pretraining stage of an MLP*. Moreover, even if the application of the KDA does not outperform the results obtained by the C-LSTM (even if the results are very close), it is worth noting that the proposed classifier is a very simple multilayered feed-forward network applied in a very informative space. Further extensions which use more complex architectures in such spaces represent an important research direction. Figure 7 shows the accuracy curves according to various approximations of the kernel space, that is, number of landmarks.

5.1.2 Semantic inferences: Argument Classification in Semantic Role Labeling

Semantic role labeling (SRL; Palmer, Gildea, and Xue 2010) consists in detecting the semantic arguments associated with the predicate of a sentence and their classification into their specific roles (Fillmore 1985). For example, given the sentence "Bring the fruit onto the dining table", the task would be to recognize the verb "bring" as evoking the BRINGING frame, with its roles, THEME for "the fruit," and GOAL for "onto the dining table". AC corresponds to the subtask of assigning labels to the sentence fragments spanning individual roles.

As proposed in Moschitti, Pighin, and Basili (2008), SRL can be modeled as a multi-classification task over each parse tree node n, where argument spans reflect sub-sentences covered by the tree rooted at n. Consistently with Croce, Moschitti, and Basili (2011), in our experiments the KDA has been empowered with an SPTK, operating over GRCT derived from dependency grammar, as shown in Figure 9.

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

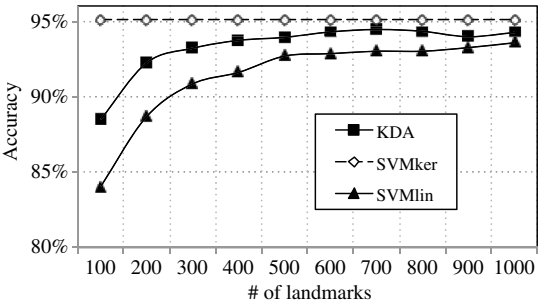

Figure 7. QC task—accuracy curves w.r.t. the number of landmarks.

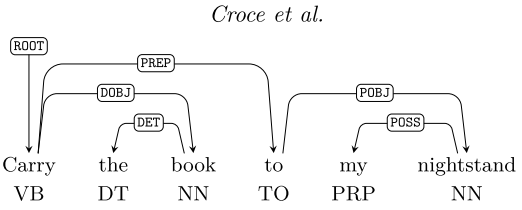

Figure 8. A dependency graph associated with "*Carry the book to my nightstand*".

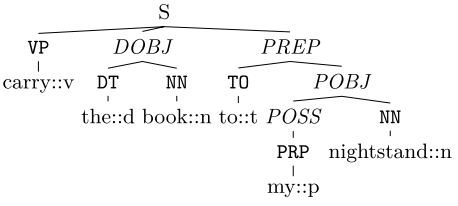

Figure 9. GRCT of the command "*Carry the book to my nightstand*".

We used the HuRIC dataset (Bastianelli *et al.* 2014; Bastianelli *et al.* 2016), including over 650 annotated transcriptions of spoken robotic commands, organized in 18 frames and about 60 arguments. We extracted single arguments from each HuRIC example, for a total of 1300 instances. We run experiments with a methodology similar to the one described in Section 5.3, but due to the limited data size we performed extensive 10-fold cross-validation, optimizing network hyperparameters via grid-search for each test set. We generated Nyström representation of a equally weighted linear combination of SPTK function with default parameters $\mu = \lambda = 0.4$ and of linear kernel function applied to sparse vector representing the instance frame. With these settings, the KDA accuracy was 96.1%.

5.2 Experimental evaluation of explanatory models

The effectiveness of the proposed approach has been measured against two different semantic processing tasks, that is, QC and AC in semantic role labeling. The KDA evaluated in Section 5.1 is here adopted and extended with LRP. For evaluating our explanation method, we defined five quality categories and associated them with values for the posteriori probability $P(C|s, e)$, as shown in Table 2. We gathered into explanation datasets hundreds of explanations from the three models for each task and presented them to a pool of annotators (further details in related subsections) for independent labeling. During the annotation process, annotators are exposed to examples classified by the KDA with an explanation and they are asked to label the explanation with one of the following classes: *Very Good* if the provided explanation is clearly convincing, *Good* if the explanation is convincing but it is not completely related to the input example so that some doubts about the system decision still remain, *Weak* if the explanation is not useful to increase the confidence of the user with respect to the system decision, *Bad* if the explanation

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

Table 2. Posterior probabilities w.r.t. quality categories

Category	$P(C\|s,e)$	$1-P(C\|s,e)$
V.Good	0.95	0.05
Good	0.8	0.2
Weak	0.5	0.5
Bad	0.2	0.8
Incoher.	0.05	0.95

Table 3. Weights for the Cohen's kappa κ_w statistics

Class	Incoher.	Bad	Weak	Good	V.Good
Incoher.	1.00	0.83	0.50	0.16	0.00
Bad	0.83	1.00	0.66	0.33	0.16
Weak	0.50	0.66	1.00	0.66	0.50
Good	0.16	0.33	0.66	1.00	0.83
V.Good	0.00	0.16	0.50	0.83	1.00

Table 4. Information gains for the three explanatory models applied to the SRL-AC and QC datasets. k_w is the weighted Cohen's kappa κ_w

	Basic	Multiplicative	Contrastive	*accuracy*	κ_w
QC	0.548	0.514	0.576	0.926	0.667
SRL-AC	0.669	0.663	0.667	0.961	0.783

makes the annotator believe that the system decision is not correct, while *Incoherent* corresponds to the case where the explanation is clearly inconsistent with the input example and suggests a clear error of the system in providing its answer. Annotators had no information of the correctness of the system emissions but just knowledge about the dataset entropy. We addressed their consensus by measuring a weighted Cohen's Kappa (adopting the weights reported in Table 3).

5.3 Evaluating question classification

We generated the Nyström representation of the CSPTK (Annesi, Croce, and Basili 2014) function with default parameters $\mu = \lambda = 0.4$. Using 500 landmarks, the KDA accuracy was 92.6%.

A group of three annotators evaluated an explanation dataset of 300 explanations (perfectly balanced between correct and not correct classification), composed of 100 explanations for each model. Performances are shown in Table 4: all three explanatory models were able to gain more than half the required information in order to ascertain the correctness of the classification. As an example, consider:

> *I think "What year did Oklahoma become a state ?" refers to a* `NUMBER` *since it reminds me of "The film Jaws was made in what year ?"*

The model provided an evidently coherent analogy, but this is a easy case due to the occurrence in both questions of very discriminative words, i.e *"what year."* However, the system is also able to capture semantic similarities when both syntactic and lexical features are different. For example:

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

I think "Where is the Mall of the America ?" refers to a `LOCATION` *since it reminds me of "What town was the setting for The Music Man ?".*

This is an high-quality explanation since the system provided an analogy with a landmark requesting the same fine-grained category but with little sharing of lexical and syntactic information (note, for example, the absence in the landmark of the very discriminative word *"where"*). Let us now consider the case of wrong classification:

I think "Mexican pesos are worth what in U.S. dollars ?" refers to a `DESCRIPTION` *since it reminds me of "What is the Bernoulli Principle ?"*

The system provided an explanation that is not possible to easily interpret: indeed, it was labeled as [Incoherent] by all the annotators.

However, the system suffers for two issues. First, in the case of negative modality and correct classification, explanations, albeit coherent, can be trivial and do not actually help in reducing uncertainty about the correct target class. For example,

I think "What is angiotensin ?" does not refer to a `NUM` *since different from "What was Einstein 's IQ ?".*

Here the explanation is correct but obvious; instead, a negative analogy with a very likely class, that is, `ENTITY` or `DESCRIPTION`, would have provided some disambiguation. Moreover, some questions are inherently ambiguous due to the lack of a broader context. This can lead to prediction errors and as well as to user actually being misled by the explanation, for example,

I think "What is the sales tax in Minnesota ?" refers to a `NUMBER` *since it reminds me of "What is the population of Mozambique ?" and does not refer to a* `ENTITY` *since different from "What is a fear of slime ?".*

In this example, the explanation makes `NUMBER` to appear as a more likely target for the question than `ENTITY`, although seemingly correct this is not the right label. Here the lack of contextual information in the question itself is the case.

5.3.1 Evaluating Argument Classification in Semantic Role Labeling

As discussed in Section 5.1.2, the KDA architecture has been successfully applied to the task of AC in semantic role labeling (Palmer *et al.* 2010), The underlying dataset is the HuRIC corpus, on which the KDA achieves 96.1% accuracy.

Among the available examples, we sampled 692 explanations equally balanced among true positives, false positives, false negatives, and true negatives. Due to the required balanced representation of all classes, the prior probability of the sample thus corresponds to an entropy of 0.998. In order to limit any bias, two annotators were exposed in a partition almost identically distributed among the three explanatory models.

Results are shown in Table 4. In this task, all models were able to gain more than two thirds of needed information. The alike scores of the three models are probably due to the narrow linguistic domain of the corpus and the well-defined semantic boundaries between the arguments.

In a scenario such as domotic Human Robotic Interfaces, the quality of individual explanatory models is very important as the robot is made capable of using explanation in a dialogue with the user. Let us consider the following examples obtained by the contrastive model:

I think **"the washer"** *is the* CONTAINING OBJECT *of* CLOSURE *in "Robot can you* OPEN ***the washer****?" since it reminds me of* **"the jar"** *in "*CLOSE ***the jar****" and it is not the* THEME *of* BRINGING *since different from* **"the jar"** *in "*TAKE ***the jar*** *to the table of the kitchen".*

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

This argumentation is very rich. It must be observed that it is not just the result of a text similarity metrics, that is, the kernel. In the example, the lexical overlap between the command and the explanation is very limited. Rather, the explanation is strictly dependent on the model and on the instance. The command cited is the activated one, that is the one that has been found useful in the inference. This is a dynamic side effect of the KDA model. It has thus a dynamic nature that changes across the different situations, that is, cases. In the situation

> *I think* **"me"** *is the* BENEFICIARY *of* BRINGING *in "I would like some cutlery can you* GET ***me*** *some?" since reminds me of* **"me"** *in "*BRING ***me*** *a fork from the press." and it is not the* COTHEME *of* COTHEME *since different from* **"me"** *in "Would you please* FOLLOW ***me*** *to the kitchen?".*

the role of grammatical information is more explicit also in the counterargument regarding the sentence *Would you please* FOLLOW ***me*** *to the kitchen?'*

Both the above commands have limited lexical overlap with the retrieved landmarks. Nevertheless, the retrieved analogies make the explanations quite effective: an explanatory model such as the contrastive one seems to successfully capture semantic and syntactic relations among input instances and closely related landmarks that are meaningful and epistemologically clear.

6. Conclusion

This paper discusses the role of semantic kernels in the definition of vector embeddings that are able to support the explanation of quantitative linguistic inferences, such as those provided by neural networks. We focused on two aspects. First, through dimensionality reduction methods we propose to use the Nyström reconstruction as an embedding method. This approach capitalizes, through the notion of kernel, the lexical semantic and grammatical knowledge implicitly represented by parse trees and dependency graphs by giving rise to meaningful vectors. Second, it has been shown how the Nyström reconstruction vectors can be straightforwardly used to compile fluent linguistic expressions that explain the inferences carried out by a trained model. By exploiting the notion of landmark, Nyström vectors map input instances into weighted linear combinations of similarity scores with the landmarks, that is, concrete and labeled examples. These examples are retrieved as input nodes activated by the network decision: they thus correspond to meaningful examples that contributed positively to the final classification or negatively with negative activations. The resulting process allows to generate epistemologically transparent and linguistically fluent explanations as combination of positively activated examples or negatively weighted counterexamples.

In this work, a novel evaluation methodology based on Information Theory is then provided to evaluate the impact of the explanations made available. In particular, performances correspond to increase in the information gain, that is decrease in entropy. Empirical investigations have been discussed for the QC and the AC tasks, which are typical examples of semantically complex inferences. The outcomes show how explanatory models provide helpful contribution to the confidence of the user in the network decision: the explanation augment confidence in correct decisions and lower down the confidence for the network errors. This clearly shows for two independent tasks that explanations are made possible over a KDA-like neural network. Given that KDA and in particular the proposed Nyström embeddings can be largely used for epistemologically clear neural learning in natural language processing, we think that they correspond to meaningful embeddings with huge potential for better neural learning models. First, they promote language semantics in a natural way and create associations between input instances and decisions that are harmonic with respect human (logical) intuition. In a sense, linguistic inferences are explained without necessarily moving out of the language level. Second, they are mathematically solid models for different levels of language semantics according to different kernel formulations. In this

https://doi.org/10.1017/S1351324919000238 Published online by Cambridge University Press

way, the embeddings can be fine tuned to tasks, without impacting on the learning architecture but only by modeling different aspects of language syntax and semantics in the kernel function. Finally, the explanations proposed in this paper correspond just to an early stage of the research. In fact, there are many ways in which activated landmarks can be made useful in the explanation process and we are in a very early stage of such an exploration. For example, argumentation theory, as applied to the landmarks active in a decision and the source input example, can provide very rich ways to compile justification, that is, short texts that argue for a decision.

References

Annesi P., Croce D. and Basili R. (2014). Semantic compositionality in tree kernels. *CIKM*. ACM.

Bach S., Binder A., Montavon G., Klauschen F., Müller K.-R., Samek W. and Suárez Ó.D. (2015). On pixel-wise explanations for nonlinear classifier decisions by layer-wise relevance propagation. *PloS One* **10**, 1–46.

Baehrens D., Schroeter T., Harmeling S., Kawanabe M., Hansen K. and Müller K.-R. (2010). How to explain individual classification decisions. *Journal of Machine Learning Research* **11**, 1803–1831.

Bastianelli E., Castellucci G., Croce D., Iocchi L., Basili R. and Nardi D. (2014). Huric: a human robot interaction corpus. *LREC*. ELRA.

Bastianelli E., Croce D., Vanzo A., Basili R. and Nardi D. (2016). A discriminative approach to grounded spoken language understanding in interactive robotics. *IJCAI*.

Bengio Y., Courville A. and Vincent P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **35**(8), 1798–1828.

Cancedda N., Gaussier É., Goutte C., and Renders J.-M. (2003). Word-sequence kernels. *Journal of Machine Learning Research* **3**, 1059–1082.

Chakraborty S., Tomsett R., Raghavendra R., Harborne D., Alzantot M., Cerutti F., Srivastava M.B., Preece A.D., Julier S.J., Rao R.M., Kelley T.D., Braines D., Sensoy M., Willis C.J. and Gurram P. (2017). Interpretability of deep learning models: A survey of results. *SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI*.

Chang C.-C. and Lin C.-J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* **2**(3), 27:1–27:27.

Collins M. and Duffy N. (2001). Convolution kernels for natural language. *NIPS* 625–632.

Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K. and Kuksa P. (2011). Natural language processing (almost) from scratch. *Journal of Artificial Intelligence Research* **12**, 2493–2537.

Cortes C. and Vapnik V. (1995). Support-vector networks. *Machine Learning* **20**(3), 273–297.

Croce D., Filice S., Castellucci G. and Basili R. (2017). Deep learning in semantic kernel spaces. *ACL*.

Croce D., Moschitti A. and Basili R. (2011). Structured lexical similarity via convolution kernels on dependency trees. *EMNLP*.

Devlin J., Chang M.-W., Lee K. and Toutanova K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv:1810.04805.

Drineas P. and Mahoney M.W. (2005). On the nyström method for approximating a gram matrix for improved kernel-based learning. *Journal of Machine Learning Research* **6**, 2153–2175.

Erhan D., Courville A. and Bengio Y. (2010). Understanding representations learned in deep architectures. Technical Report 1355, Montreal, QC, Canada: Université de Montréal/DIRO.

Faruqui M., Tsvetkov Y., Yogatama D., Dyer C. and Smith N.A. (2015). Sparse overcomplete word vector representations. *ACL-IJCNLP*.

Filice S., Castellucci G., Croce D. and Basili R. (2015). Kelp: a kernel-based learning platform for natural language processing. *ACL System Demonstrations*. **1**, 19–24.

Filice S., Castellucci G., Martino G.D.S., Moschitti A., Croce D., and Basili R. (2018). Kelp: a kernel-based learning platform. *Journal of Machine Learning Research* **18**(191), 1–5.

Fillmore C.J. (1985). Frames and the semantics of understanding. *Quaderni di Semantica* **6**(2).

Frosst N. and Hinton G. (2017). Distilling a neural network into a soft decision. *Proceedings of the First International Workshop on Comprehensibility and Explanation in AI and ML 2017 co-located with 16th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2017)*, Bari, Italy, November 16th and 17th, 2017.

Goldberg Y. (2016). A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research* **57**, 56–65.

Hochreiter S. and Schmidhuber J. (1997). Long short-term memory. *Neural Computation* **9**(8), 1735–1780.

Hsieh C.-J., Chang K.-W., Lin C.-J., Keerthi S.S. and Sundararajan S. (2008). A dual coordinate descent method for large-scale linear svm. *ICML*. ACM.

Jacovi A., Sar Shalom O. and Goldberg Y. (2018). Understanding convolutional neural networks for text classification. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. ACL.

Kim Y. (2014). Convolutional neural networks for sentence classification. *EMNLP*.
Kononenko I. and Bratko I. (1991). Information-based evaluation criterion for classifier's performance. *Machine Learning* **6**(1), 67–80.
Lei T., Barzilay R. and Jaakkola T. (2016). Rationalizing neural predictions. *EMNLP*. ACL.
Li X. and Roth D. (2006). Learning question classifiers: the role of semantic information. *Natural Language Engineering* **12**(3), 229–249.
Lipton Z.C. (2018). The mythos of model interpretability. *Queue* **16**(3), 30:31–30:57.
Manning C.D., Surdeanu M., Bauer J., Finkel J., Bethard S.J. and McClosky D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Baltimore, Maryland. pp. 55–60.
Mikolov T., Chen K., Corrado G. and Dean J. (2013). Efficient estimation of word representations in vector space. *CoRR abs/1301.3781*.
Mitchell J. and Lapata M. (2010). Composition in distributional models of semantics. *Cognitive Science* **34**(8), 161–199.
Moschitti A. (2006). Efficient convolution kernels for dependency and constituent syntactic trees. *ECML*.
Moschitti A. (2012). State-of-the-art kernels for natural language processing. *ACL (Tutorial Abstracts)*. Association for Computational Linguistics, p. 2.
Moschitti A., Pighin D. and Basili R. (2008). Tree kernels for semantic role labeling. *Computational Linguistics* **34**, 193–224.
Padó S. and Lapata M. (2007). Dependency-based construction of semantic space models. *Computational Linguistics* **33**(2), 161–199.
Palmer M., Gildea D. and Xue N. (2010). *Semantic Role Labeling*. IEEE Morgan & Claypool Synthesis eBooks Library. San Rafael, CA, USA: Morgan & Claypool Publishers.
Pennington J., Socher R. and Manning C.D. (2014). Glove: Global vectors for word representation. *EMNLP*.
Peters M.E., Neumann M., Iyyer M., Gardner M., Clark C., Lee K. and Zettlemoyer L. (2018). Deep contextualized word representations. *NAACL*.
Ribeiro M.T., Singh S. and Guestrin C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. *CoRR abs/1602.04938*.
Robert Müller K., Mika S., Rätsch G., Tsuda K. and Schölkopf B. (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks* **12**(2), 181–201.
Sahlgren M. (2006). *The Word-Space Model*. PhD Thesis, Stockholm University.
Schütze H. (1993). Word space. *Advances in Neural Information Processing Systems*, Vol. 5. Burlington, MA, USA: Morgan-Kaufmann.
Shawe-Taylor J. and Cristianini N. (2004). *Kernel Methods for Pattern Analysis*. New York, NY, USA: Cambridge University Press.
Simonyan K., Vedaldi A. and Zisserman A. (2013). Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR abs/1312.6034*.
Socher R., Perelygin A., Wu J., Chuang J., Manning C.D., Ng A. and Potts C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. *EMNLP*.
Spinks G. and Moens M.-F. (2018). Evaluating textual representations through image generation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. ACL.
Strubell E., Verga P. andor D., Weiss D. and McCallum A. (2018). Linguistically-informed self-attention for semantic role labeling. *EMNLP*.
Subramanian A., Pruthi D., Jhamtani H., Berg-Kirkpatrick T. and Hovy E.H. (2018). Spine: Sparse interpretable neural embeddings. *AAAI*.
Tai K.S., Socher R. and Manning C.D. (2015). Improved semantic representations from tree-structured long short-term memory networks. *ACL-IJCNLP*.
Trifonov V., Ganea O.-E., Potapenko A. and Hofmann T. (2018). Learning and evaluating sparse interpretable sentence embeddings. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.
Vapnik V.N. (1998). *Statistical Learning Theory*. New York, NY, USA: Wiley-Interscience.
Walton D., Reed C. and Macagno F. (2008). *Argumentation Schemes*. Cambridge, England, UK: Cambridge University Press.
Williams C. K.I. and Seeger M. (2001). Using the Nyström method to speed up kernel machines. *NIPS*.
Zeiler M.D. and Fergus R. (2013). Visualizing and understanding convolutional networks. *CoRR abs/1311.2901*.
Zhang R., Lee H. and Radev D.R. (2016). Dependency sensitive convolutional neural networks for modeling sentences and documents. *NAACL-HLT*.
Zhou C., Sun C., Liu Z. and Lau F.C.M. (2015). A C-LSTM neural network for text classification. *CoRR abs/1511.08630*.

Cite this article: Danilo C, Rossini D and Basili R. Neural embeddings: accurate and readable inferences based on semantic kernels. *Natural Language Engineering* **25**, 519–541. https://doi.org/10.1017/S1351324919000238