DeepKAF: A Heterogeneous CBR & Deep Learning Approach for NLP Prototyping

Kareem Amin
Smart Data and Knowledge Services
German Research Center for Artificial Intelligence
Technische Universität Kaiserslautern
Kaiserslautern, Germany
kareem.amin@dfki.uni-kl.de

Stelios Kapetanakis
School of Computing Engineering and Mathematics
University of Brighton
Brighton, United Kingdom
s.kapetanakis@brighton.ac.uk

Nikolaos Polatidis
School of Computing Engineering and Mathematics
University of Brighton
Brighton, United Kingdom
n.polatidis@brighton.ac.uk

Klaus-Dieter Althoff
Intelligent Information Systems Lab
Institute of Computer Science
University of Hildesheim
Hildesheim, Germany
klaus-dieter.althoff@dfki.uni-kl.de

Andreas Dengel
Smart Data and Knowledge Services
German Research Center for Artificial Intelligence
Technische Universität Kaiserslautern)
Kaiserslautern, Germany
andreas.dengel@dfki.uni-kl.de

***Abstract*—With widespread modernization, digitization and transformations of most of industries, Artificial Intelligence (AI) has become the key enabler in that modernization journey. AI offers substantial capabilities to solve new problems and optimise existing solutions specialising on specific problems and learning from different domains. AI solutions can be either explainable or black box ones with the latter being urged to improve since they cannot trust. Case-based Reasoning (CBR) is an explainable AI approach where solutions are provided along with relevant explanations in terms of why a solution was selected. However, CBR, like most other explainable approaches, has several limitations in terms of scalability, large data volumes, domain complexity, that reduce its ability to scale any CBR system in industrial applications. In this paper, we provide a heterogeneous CBR framework - DeepKAF where we combine CBR paradigm with Deep Learning architectures to solve complicated Natural Language Processing (NLP) problems (eg. mixed language and grammatically incorrect text).DeepKAF is built based on continuous research in the area of Deep Learning and CBR. DeepKAF has been implemented and used across different domains, test use cases and research models as an ensemble deep learning and CBR Architecture.**

***Index Terms*—Deep Learning, Case-based Reasoning, Natural Language Processing**

I. Introduction

Decisions under pressure is a common characteristic of modern processes and real time applications. With a constant flow of valuable data points, processes are called to perform better and faster whilst the decision time remains the same if not shorter. Over the past years we have seen impressive work in the areas of Neural Networks and Deep Learning enabling faster reasoning over constantly working data flows, however there is little work in the areas of reasoning under fuzziness, incomplete information and uncertainty. Decision Support Systems that deal with customers based on text information have a high degree of fuzziness and uncertainty.This situation becomes even more challenging when their available text contains abbreviations, missing sentences or is multilingual. Motivated from the above challenges this work presents Deep Knowledge Acquisition Framework (DeepKAF) a Case-based Reasoning Framework for rapid application prototyping on Natural Language Processing Workflows. DeepKAF presents an end-to-end natural language framework and it has remarkable applicability and acceptance in industrial applications that share multi-languages, mixed notations and content fuzziness. The key idea behind its inception was the observation that processes can be represented as workflows and workflow states can be inferred from text flows (signals) using historical knowledge (cases). Workflow signals can be inferred to workflow processes [21], [22] and processes to experiences leading to appropriate mitigation policies. This work has been evolved over a number of real applications [5], [6], [4] and this work presents its core concepts, architecture and implementation using deep learning models and similarity metrics. Our contributions in this work are:

1) DeepKAF Framework as an NLP workflow representation framework using CBR
2) Its architecture and Implementation
3) An Evaluation using real time NLP workflow data

The structure of this paper is as follows: Section II presents DeepKAF at high level, its architecture and its modelling and encoding components, Section II covers Feedback and Modelling Retraining, focusing on its recent novelties in terms of Word embeddings and the addition of Siamese Networks,

978-1-7281-6799-2/20/$31.00 ©2020 IEEE

Section IV presents its evaluation so far using Historical data and a variety of methods to provide results. Finally, Section V concludes this work and presents the forthcoming additions and upcoming improvements to this framework.

II. Deep Knowledge Acquisition Framework (DeepKAF)

DeepKAF framework has been designed and implemented to leverage Case-based Reasoning as an explainable AI approach by using various Deep Learning models to overcome and mitigate the limitations of CBR implementations. According to the yearly survey about what's hot in CBR domain [2], Case acquisition from raw data, including text and diagrams is on the top of the list along with implementing CBR approaches that can deal with high velocity and veracity data volumes. **DeepKAF** is defining the procedure of implementing a CBR system and inject Deep Learning models while still being able to provide adequate level of explainability (See Figure 1). Within specific industrial domains where tacit knowledge is present, deep learning approaches do not suffice and specialist input is required to mitigate specialised needs. **DeepKAF** amplifies case-based reasoning with deep learning methods to cope with specialist domain challenges pertaining to large text corpuses, multi language content and mixed data types e.g. numeric, catecorigal, etc..

DeepKAF is mainly improving two key elements of CBR components, these of: 1. Similarity Measures 2. Retrieval

In the next sections, we will explain **DeepKAF** in detail by showing the different similarity measures and retrieval components.

A. Building Similarity Measures using Deep Learning Models Components

DeepKAF similarity measures component comprise three key models: 1. Autoencoders 2. Word Embeddings 3. Siamese Networks Within any CBR system, building similarity measures are considered one of the most tedious and complicated task [12] [10] [13]. Similarity measures are based on specialised domain knowledge and can identify how close cases are to each other. To decode domain knowledge and describe how cases and attributes are related to each other, intensive domain expertise is required. In **DeepKAF**, the main goal is to semi-automate the building of similarity measures in Textual Case-based Systems.

Figure 2 depicts the process of building similarity measures with CBR systems using Autoencoders, Word Embeddings, and Siamese Neural Network.

Step 1: Use denoising and dimentiality reduction autoencoders on the input data.
Step 2: Build Word Embeddings for the text.
Step 3: Split the dataset into Training/Test Data.
Step 4: Train the Siamese Neural Networks.
Step 5: Check the accuracy with the test data.
Step 6: If the accuracy is not high, do some hyper-parameters tuning and re-run the entire process.

a) Autoencoders: Autoencoders are self-supervised neural networks designed primarily for dimensionality reduction purposes [14]. Autoencoders can unify content in isometric vector representations and reduce and represent input with fewer features. **DeepKAF** uses Autoencoders to reduce manual tagging from domain experts and accelerate the creation of traditional CBR cases. In **DeepKAF** [8], Autoencoders have been used successfully to denoise and summarize input text prior feeding it to a novel Siamese network for similarity calculations during DeepKAF′s training phase as well as find similarities during the retrieval phase.

b) Word Embeddings: For more clarification, word embedding is the method of translating the text into a numerical representation (Vectorization). In **DeepKAF** architecture, the Word Embeddings concept is used intensively. Mitolov et al in 2013 have brought the NLP group to Word2Vec [16]. Word2vec comprises two language models in the neural network: a Continuous Bag of Words (CBOW) and Skip-gram(SG). For both CBOW and SG, a predefined-length text window is moved around the corpus and the network is trained in an iterative way with the words within the window. While the CBOW model is trained to predict a word based on the corresponding terms in the centre of a pre-defined window, the Skip-gram model training aims to predict meanings based on the concept of the main phrase. The model learns a linear transformation which is the hidden layer and is considered as the word representation. DeepKaf uses SG since it is superior performance-wise in the recognition of semantic tasks [9].

c) Siamese Networks: Siamese Neural Networks (SNNs) usually contain several sub neural-networks that share the same parameter and weight configuration. SNNs are known for their ability to distinguish correlations or associations among distinct objects. Usually distinct sub-networks are used to handle related inputs, while the meta-processing node must merges their outputs into a decisive one network [15]. DeepKAF uses SNNs since they have the following characteristics:

1) SNNs are more solid compared to Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) whilst appying them to advanced text tasks
2) SNNs provide robust and more meaningful text representation (embeddings)
3) Sub-networks have collocated weights, having as a result a reduction in the number of learning parameters, thus reduced data requirements and less propensity to overfit.

B. Retrieval Component

In the retrieval component, the models that have been built and trained are going to be used in order to clean the input text first and then retrieve the most similar cases via the trained Siamese Network. The siamese network gives a range between 0 (no similarity) and 1 (identical) to how similar the cases are. Then according to the domain the CBR system is working in, the retrieved results should be sorted based on the global and local similarity measures.

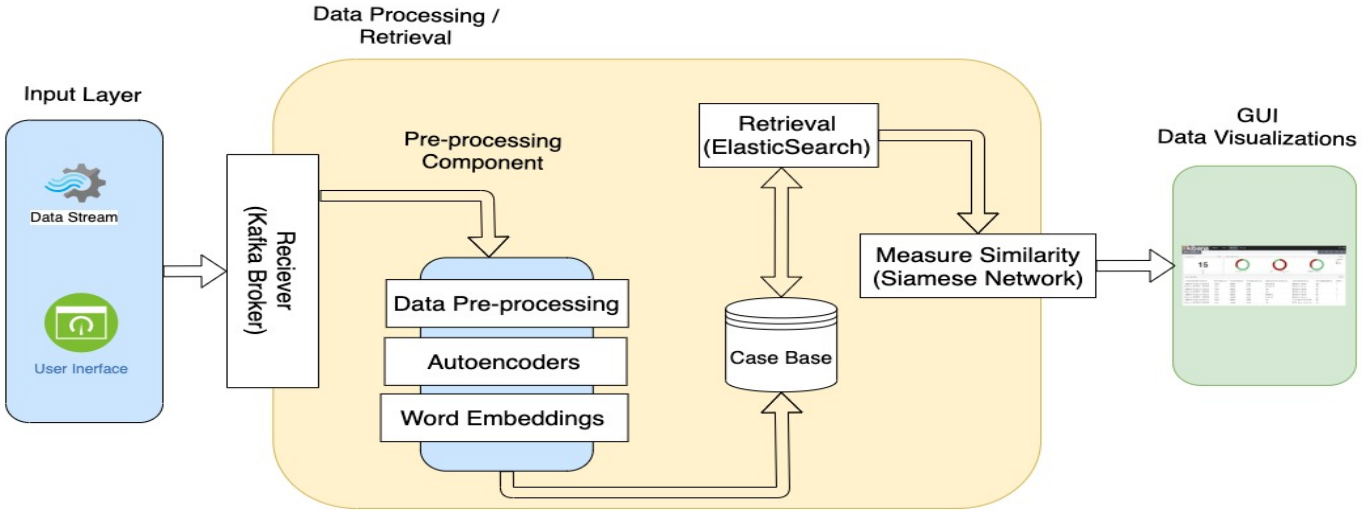

Fig. 1. DeepKAF Architecture

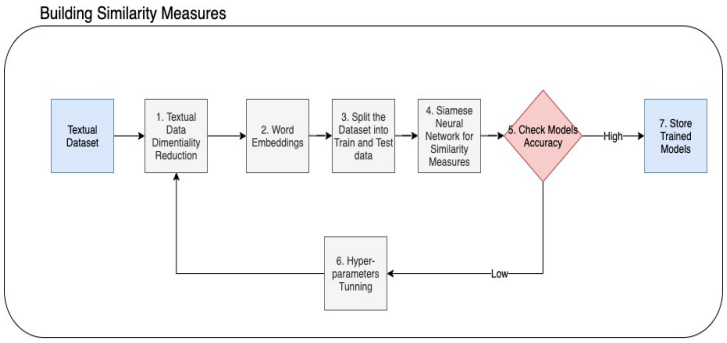

Fig. 2. Building Similarity Measures Process

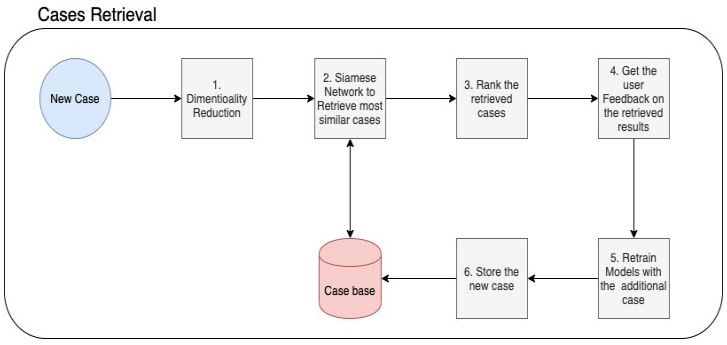

Fig. 3. Retrieval Process

Figure 3 shows the retrieval process within the CBR paradigm combined with the trained deep learning models. Traditional CBR uses similarity knowledge for retrievals (similarity-based retrievals-SBR) [7]. In DeepKAF, the traditional similarity matrix is replaced by the trained models that can give degrees of similarities between two distinctive textual cases.

The Retrieval Process is as the following:

Step 1: Use denoising and dimentiality reduction autoencoders on the input data (exactly as in the building similarity measures phase).
Step 2: Use the trained siamese network to retrieve the nearest neighbours.
Step 3: Rank the neighbour cases according to the domain preferences.
Step 4: Get the end user feedback on the retrieved results.
Step 5: Retrain the models according to the methodologies described in **"Section III"**
Step 6: Store new case in the knowledge base along with its associated potential solutions.

III. Feedback and Models Retraining

A mean to provide feedback and "back-input" is an essential process for any industrial CBR system. Withing a traditional CBR cycle, constant user feedback is used for this purpose. Typically a new case and its solution are added to the Knowledge Base as part of the "lazy" learning process that CBR systems use to improve. For **DeepKAF** the feedback process leads to ongoing deep learning model retraining that extends the used vector space. This is being used respectively to measure similarities more accurately and increase the coverage of word embeddings model by increasing its vocabulary.

In this section, we will describe the available re-training approaches that can be used within DeepKAF.

A. Word Embeddings Model

DeepKAF uses a word embedding constructed during its training phase, which is provided from the available case content, vocabulary and metadata. Any new case provides acts as feedback input, having as a result the retraining of the existing word embeddings model on new terms, available words and sentences. Several word embeddings models have been used throughout **DeepKAF's** [6], with no specific model

prevailing and thus recommended over another other. An observation from the authors is that embeddings are affected from the type of text, the training which should be based on high coverage of available words and sentences for a highly relevant embeddings model to be selected. There are two main retraining approaches on this:

a) : Incremental training techniques differ based on the model to be retrained. **DeepKAF** has tested several word embedding models among which word2vec [16] and its descendants (eg. sentence2vec, sentence2sentence etc...)seemed most applicable. Word2vec models do not offer an option for direct, incremental training. They have to be re-implemented extensively and heavily modified -from source code- to be able to build a model that can be retrained.

b) : Transfer learning is the higher degree of improved learning related to a new task. Transfer learning is achieved through existing knowledge acquisition from a similar or historical tasks that have already contributed to knowledge acquisition [19]. In **DeepKAF**, transfer learning has been used in retraining a word2vec model with new words deriving from the new cases. Word Embeddings have been a key component in the success of the entire approach because they are used by a full range of the neural network topologies to measure similarities like LSTMs and MaLSTM that will be presented in the following sections. **DeepKAF** delegates the training to a new model by propagating back to word vectors (word vector inputs can be fixed in which case the issue below does not apply) If the LSTM training corpus is sufficiently large such as that the vocabulary in training phase retrains all the word2vec vectors, this is regarded as successful retraining phase. If the training corpus does not retrain all vectors, and we get a word in testing phase, that has a word that was not retrained, there is a risk of lowering the model accuracy, since untrained vectors will be mixed along with the ones that were trained. This specific case requires extra attention. In summary, if word vectors are chosen for transfer learning, and the word vectors are retrained in the new space, the framework needs to ensure the training moves all the words in the original vector space - that is the new corpus should have all the words in the original word2vec vector output.

B. Siamese Networks

Unlike other Neural Networs that learn to predict over different classes, Siamese Neural Networks (SNNs) can learn how similar or dissimilar two objects are. Hence, with every new case, the SNN model needs to be retrained on the whole dataset which is an exhaustive task in terms of time and processing power. Based on several experiments to establish the most appropriate similarity degree it was observed that the top 10 retrieved results (See section ”IV Experimental Results”) can be used in data augmentation techniques to reproduce a small dataset that consists of similar objects and use transfer learning for the retraining process.

IV. Experimental Results

DeepKAF is a result of a ongoing research throughout a period of four years and the work progress has been published in several conferences [6] [5] [4] [8]. **DeepKAF** has showed very promising results when applied in the industrial domain. This section presents, explains and evaluates the key use case we have implemented. The results shown below are the final results with the best fitting models and hyper-parameters.

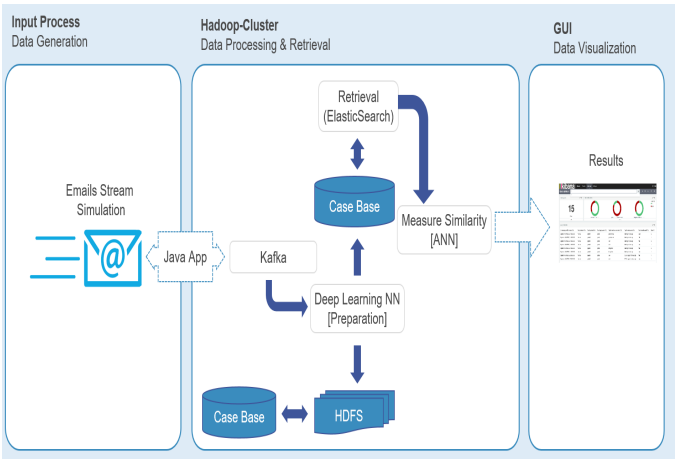

Fig. 4. Technical DeepTMS Architecture

A. DeepTMS - Ticket Management System Use Case

For the evaluation purposes we used a ticket management system (TMS) using an ensemble CBR approach based on combining CBR with Deep Neural Networks and Data Streaming Technologies. The aim was to increase the accuracy and ”response time” to ”incoming” support tickets. This was achieved by identifying and supplying the most relevant cases and their solutions from closed, neighbour issues using a historical knowledge-base. Any retrieved cases/solutions were identified and proposed to a Help/ Support engineer in real-time support cases. The research on **DeepKAF** has been thoroughly carried on using different DL models to enhance the consistency as well as the functionality of the framework while still minimizing the efforts to acquire new knowledge [5] [4] [6] [3] [8]. The provided application is evaluated based on its capability to provide better solutions to human judgement. A further metric to its capability is to be able to provide several options even to cases that no solutions would be suggested from human agents. The application and any data used for this work were based on collaborative efforts between the German Research Center for Artificial Intelligence (**DFKI**) and a Private purpose high tech engineering stakeholder.

Cases Representation: Past scenarios / cases were defined as: Problem Definition, Used Solution, Relevant Section/Department, Case Ranking/Classification and relevant Interest Tags/Keywords. A deep dive analysis into the historical cases showed a variety of difficulties, such as: cases were written in multi languages, there were several phrases with substantial grammar, syntax and semantic errors, and a large number of technical abbreviations and domain jargon. The evaluated system reflected the most common problems associated with help-desk management systems that are:

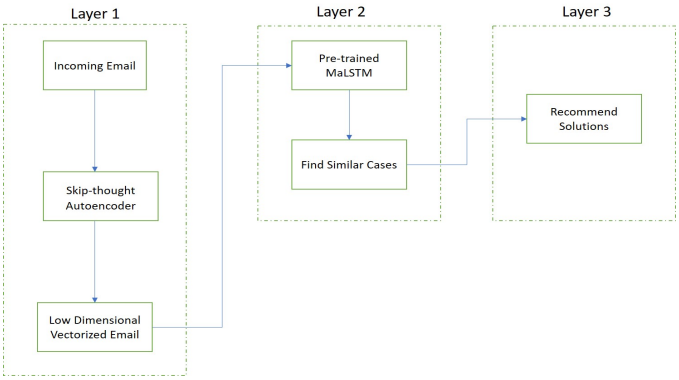

Fig. 5. DeepTMS - Ticket Management System Block Diagram

1) High velocity of requests in a variety of formats, multi channels and different regions
2) Sensitive and restrictive Service Level Agreements (SLAs) that restrict support units to mitigate requests in a limited amount of time conforming to the company commitments
3) Restrictive ticket resolution workflows that may lead to SLA infringement
4) Considerable variations in finding an effective solution, by usually ignoring or finding difficult to utilise historical knowledge. Throughout the implementation, several support practitioners were interviewed and acknowledged difficulty for cases to converge and prove useful to currently investigated cases.

The primary objective of the evaluation is to prove the efficiency of the framework using text denoising based on work previously introduced in **DeepKAF** [6]. In this work, A **Skip-thought** autoencoder was used as a distributed sentence encoder-decoder to create a vectorized low-dimensional representation of sentences. The encoded sentences were then analyzed by the pre-trained Manhattan LSTM (**MaLSTM**) [11] siamese neural networks were then used to compare and identify the most related cases in order to be able to suggest more correct solutions (see **Figure 5**).

1) Training:

Throughout the training process, three models have been used. Each model applied different learning approach varied from Supervised, semi-supervised and unsupervised.

1) **Skip-thought Autoencoder**: An Autoencoder was trained using more than 300,000 support Tickets. The aim of this step was to identify sufficient low-level representation for any used text corpus in the form of support tickets.
2) **Word2Vec**: The Word2Vec low lebel text representation model was trained using unsupervised learning. This step created sufficient word relevance and relationships among the existing word corpus [6].
3) **Siamese MaLSTM**: The last step involved the training of a fully supervised network using sentence pairs. This step was used correlate and identify semantic similarities among different sentences [4]. This step was particularly useful since it provided a numerical degree of similarity as an output.

Optimizing a deep learning model is highly demanding when training and improving a NN model. This task is usually referred as hyper-parameter parameterisation defining the structure of the network, how it is built and trained. After several iterations of hyper-parameters combination settings and based on previous research on **DeepKAF** the optimal hyper-parameters configurations that led to best performance are described and shown in **Table I**:

TABLE I
MODELS HYPER-PARAMETERS

Word2Vec	**MaLSTM**	**Skip-thought Autoencoder**
Vector Dimensionality = 150 **Window** = 10 **Min Word Freq** = 10 **Epochs** = 34 **Learning Rate** = 0.0001 **Activ Function** = SOFTMAX	**Neuron Activation Function** = Relu **Learning Rate** = 0.0001 **Regularization** = L2 (.001) **LSTM Layer Size** = 200 **Batch Size** = 64 **Epochs** = 47 **Output Layer Loss Function** = MCXENT **Activation Function** = SOFTMAX	**Embedding Dimension** = 150 **Latent Dimension** = 128 **Batch Size** = 64 **Epochs=** 76 **Optimizer** = Nadam **Learning Rate** = 0.0001 **Decoder Activation Function** = SOFTMAX

2) Results: Evaluation success was defined by offering the capability to provide better solutions against human engineers across a range of the top ten suggested solutions. Similarly the system was evaluated on the possibility to offer possible solutions in cases where no obvious solution was present. To evaluate this appropriately streams of potential support cases were simulated based on historical data and similarly to the original cases they required an appropriate case handling aca solution. Table II shows comparative columns of the obtained results between the original approach (**MaLSTM**) and the newly proposed approach **Skip-thought + MaLSTM**. The final evaluation was conducting using industry domain experts support specialists and domain skilled engineers. **DeepKAF** proposed several solutions for each ticket that were also presented to the experts to rank and evaluate the suggested outcome.

From the evaluation it can be seen that **Skip-thought** increased **DeepKAF**'s overall accuracy by selecting the best ticket solutions among a range of several solutions as well as reduced the number of not listed solutions. The evaluation stakeholders have stated that the proposed solutions were appropriate since a human support agent is also expected to find a solution within the first 10 suggested ones. Nonetheless, the key benefit of using Autoencoder models to reduce the noise across data led to a reduction in training effort compared

TABLE II
DEEPTMS - TICKET MANAGEMENT SYSTEM RETRIEVAL RESULTS

	MaLSTM		Skip-thought + MaLSTM	
Level	**Cases**	**Percentage**	**Cases**	**Percentage**
Very Relevant	8354	83.54%	8832	88.32%
Relevant	1510	15.1%	942	9.42%
Slightly Relevant	30	0.3%	215	2.15%
Not Listed	106	1.06%	11	0.11%

to literature approaches using LSTM models. An LSTM model requires substantial text training , with with both relevant and irrelevant text sections to obtain the best text representation for any given text content. Contrary to LSTM the AE training process did not require such training or similar effort since it requires training across any relevant text corpus via a self-supervised learning process. As a result AE reduce substantially the required time for overall model training.

V. CONCLUSION & FUTURE WORK

While **DeepKAF** is mainly built and experimented on textual data, we beilive that it can be applied to more complicated hetergenous data sources, like mixed textual and images data. With **DeepKAF**, CBR systems can touch new grounds when it comes to industrial implementations where data is unstructured and heterogeneous.

In this paper we wrapped up the entire processes that construct **DeepKAF** and showed how Deep Learning models could be injected within the CBR paradigm without hiding the explainability advantage of any CBR system. **DeepKAF** has been applied on a ticketing management system and was able to achieve a significant improvement of the retrieved results as per the domain experts along with minimizing the required efforts to build such a system. We have demonstrated different deep learning models that are being used and experimented that they give the best results (word2vec + Skip-Thought + MaLSTM).

One of the drawbacks of traditional word vectors is that they presume the meaning of a word is fairly constant across sentences, whereas it is common for the same word to have different meanings depending on where it is used. Hence, Deep Bidirectional Transformers should be experimented against and benchmarked against the word embedding model as a future work.

Transformers (Attention) can be used in building the word embeddings and siamese network. Transformers are a deep learning model that do vector transformation of type A into vector of type B. Transformers do not use any recurrent networks (GRU, LSTM, etc) [17]. They depend on the attention mechanism which allows each processed word at each step to have information about the other words around it. The power of transformers is that they allow parallel training of the input sequence which is very valuable.

The key strength of Transformer models is actually that they allow shorter training time compared to conventional embedding models. Bidirectional Encoder Representations **BERT** [20] transformers is an innovative technique for NLP problems. **BERT** uses bidirectional attention transformer system which can have a major impact on the lack of training data or database servers.

ACKNOWLEDGMENT

REFERENCES

[1] Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 1(7) (March 1994)

[2] Ashok K. Goel, Belen Diaz-Agudo, What's Hot in Case-Based Reasoning, Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

[3] Amin K., Lancaster G., Kapetanakis S., Althoff KD., Dengel A., Petridis M., Case-based Reasoning in Natural Language Processing: Word2vec VS fastText, 23rd UK Symposium on Case-Based Reasoning. Cambridge

[4] Amin K., Lancaster G., Kapetanakis S., Althoff KD., Dengel A., Petridis M., (2019) Advanced similarity measures using WordEmbeddings and Siamese Networks in CBR. In. Intelligent Systems and Applications. IntelliSys 2019. Advances in Intelligent Systems and Computing.

[5] Amin K., Kapetanakis S., Althoff KD., Dengel A., Petridis M., Dynamic process workflow routing using Deep Learning, AI-2018 Thirty-eighth SGAI International Conference on Artificial Intelligence. Cambridge

[6] Amin K., Kapetanakis S., Althoff KD., Dengel A., Petridis M. (2018) Answering with Cases: A CBR Approach to Deep Learning. In: Cox M., Funk P., Begum S. (eds) Case-Based Reasoning Research and Development. ICCBR 2018. Lecture Notes in Computer Science, vol 11156. Springer, Cham

[7] Kang YB., Krishnaswamy S., Zaslavsky A. (2011) Retrieval in CBR Using a Combination of Similarity and Association Knowledge. In: Tang J., King I., Chen L., Wang J. (eds) Advanced Data Mining and Applications. ADMA 2011. Lecture Notes in Computer Science, vol 7120. Springer, Berlin, Heidelberg

[8] Kareem Amin, Stelios Kapetanakis, Klaus-Dieter Althoff, Andreas Denger, Miltos Petridis "Cases without Borders: Automating Knowledge Acquisition Approach using Deep Autoencoders and Siamese Networks in Case-Based Reasoning," 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI), Portland, OR, USA, 2019, pp. 133-140

[9] Altszyler, Edgar and Sigman, Mariano and Fernández Slezak, Diego, Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database, 2016

[10] Reuss P., Witzke C., Althoff KD. (2017) Dependency Modeling for Knowledge Maintenance in Distributed CBR Systems. In: Aha D., Lieber J. (eds) Case-Based Reasoning Research and Development. ICCBR 2017. Lecture Notes in Computer Science, vol 10339. Springer, Cham

[11] Mueller, J., Thyagarajan, A. (2016) Siamese Recurrent Architectures for Learning Sentence Similarity, Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16), 2016

[12] Stram R., Reuss P., Althoff KD. (2017) Weighted One Mode Projection of a Bipartite Graph as a Local Similarity Measure. In: Aha D., Lieber J. (eds) Case-Based Reasoning Research and Development. ICCBR 2017. Lecture Notes in Computer Science, vol 10339. Springer, Cham

[13] Reuss P. et al. (2016) FEATURE-TAK - Framework for Extraction, Analysis, and Transformation of Unstructured Textual Aircraft Knowledge. In: Goel A., Díaz-Agudo M., Roth-Berghofer T. (eds) Case-Based Reasoning Research and Development. ICCBR 2016. Lecture Notes in Computer Science, vol 9969. Springer, Cham

[14] Vincent, Pascal; Larochelle, Hugo (2010). "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion". Journal of Machine Learning Research. 11: 3371–3408.

[15] Jane Bromley et al, Signature Verification using a "Siamese" Time Delay Neural Network, NIPS'93 Proceedings of the 6th International Conference on Neural Information Processing Systems Pages 737-744, 1993

[16] Tomas Mikolov and Kai Chen and Greg Corrado and Jeffrey Dean, Efficient Estimation of Word Representations in Vector Space, NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, 2013
[17] Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkoreit and Llion Jones and Aidan N. Gomez and Lukasz Kaiser and Illia Polosukhin, Attention is All you Need, ArXiv, 2017
[18] Kiros, R., Zhu, Y., Salakhutdinov, R.R., Zemel, R.S., Torralba, A., Urtasun, R., & Fidler, S. (2015). Skip-Thought Vectors. NIPS.
[19] Lyan Verwimp and Jerome R. Bellegarda, Reverse Transfer Learning: Can Word Embeddings Trained for Different NLP Tasks Improve Neural Language Models?, 2019, arXiv
[20] Jacob Devlin and Ming-Wei Chang and Kenton Lee and Kristina Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, dblp computer science bibliography, https://dblp.org, 2018
[21] Stelios Kapetanakis, Miltos Petridis, Brian Knight, Jixin Ma, Liz Bacon, A case based reasoning approach for the monitoring of business workflows, International Conference on Case-Based Reasoning, 390-405
[22] Stelios Kapetanakis, Miltos Petridis, Evaluating a Case-Based Reasoning Architecture for the Intelligent Monitoring of Business Workflows, Successful Case-based Reasoning Applications-2, 43-54