Evaluating Explainable Career Path Predictions using Domain Experts

Roan Schellingerhout[1,2*], Volodymyr Medentsiy[2], Nava Tintarev[1], Francesco Barile[1] and Maarten Marx[3]

[1*]Department of Advanced Computing Sciences, Maastricht University, Paul-Henri Spaaklaan 1, Maastricht, 6229 EN, The Netherlands.

[2]Randstad Groep Nederland, Diemermere 25, Diemen, 1112 TC, The Netherlands.

[3]IRLab informatics institute, University of Amsterdam, Science Park 904, Amsterdam, 1098 XH, The Netherlands.

*Corresponding author(s). E-mail(s): roan.schellingerhout@maastrichtuniversity.nl;
Contributing authors: volodymyr.medentsiy@randstadgroep.nl; n.tintarev@maastrichtuniversity.nl; f.barile@maastrichtuniversity.nl; m.j.marx@uva.nl;

Abstract

Career path prediction aims to determine a potential employee's next job, based on the jobs they have had until now. While good performance on this task has been achieved in recent years, the models making career predictions often function as black boxes. By integrating components of explainable artificial intelligence (XAI), this paper aims to make these predictions explainable and understandable. To study the effects of explainability on performance, three non-explainable state-of-the-art baselines were compared to three similar, but explainable, alternatives. Furthermore, user studies were performed with recruiters to determine the plausibility and usefulness of the explanations generated by the models. Results show that the explainable alternatives perform on-par with their non-explainable counterparts. In addition, the explainable models were determined to provide reasonable and understandable explanations by recruiters.

Keywords: Career path prediction, Explainable AI, Neural networks, User studies

1 Introduction

With the rise of short-term contracts, it has become more difficult for job seekers to find stable positions of employment [1]. In addition, due to the average education level of the workforce having increased considerably in recent years, potential employees are faced with more opportunities than ever before [2]. This has made it significantly more difficult for job seekers to find positions that fit their needs, and for employment agencies to match candidates and vacancies. To assist in this complex and challenging task, many employment agencies have started to use computer-aided HR matchmaking using machine learning to find suitable positions for individuals, and capable employees for companies [3]. This task is called *career path prediction*, which aims to predict a person's next position of employment, given their career up until this point.

Previous research on automated career path prediction tends to share a common flaw: a lack of explainability [4–7]. While deep learning models can make highly accurate predictions, they often function as a black box. Although good results that are difficult to interpret are acceptable in many use cases, choosing a new career is such an impactful event in a person's life that it is unrealistic to expect candidates to blindly trust the predictions. This is why explainability is such a crucial requirement for career path prediction models. Through the use of explainable artificial intelligence (XAI) [8], individuals with little knowledge of deep learning (e.g. recruiters or job seekers themselves) are able to interpret to what extent, and in what ways, each variable contributed to the final outcome of the model. By being able to concretely determine *why* a given position is ideal for a person, the recommendation becomes considerably more transparent, understandable, and thus more trustworthy [9].

In this work, when we refer to *explanations*, we primarily refer to the second sense identified in the concise Oxford dictionary: "give a reason or justification for". More specifically, in this paper we check whether the justification given by the system reflects the justifications given by experts in the recruitment domain.

Although previous works within the field of career path prediction have not considered explainability, research on similar problems has done so extensively. Most popularly, within the field of financial forecasting, another time series classification task, explainable AI has been on the rise. The two most common techniques used in the literature for explaining financial forecasting are LIME (Local Interpretable Model-Agnostic Explanations) [10] and SHapley Additive exPlanations (SHAP) [11]. These techniques are generally valid in this context because they explain an opaque model's decision locally or globally, giving insights into the features (i.e., technical indicators, stock-related news, buy/trigger signals, etc.) that contributed to the model's outputs. For example, the authors of [12] and [13] created an interactive dashboard for price prediction movements based on time series and integrated it with LIME explanations on the stock-related news to trigger buy or sell

signals. Further, Benhamou et al. [14] used SHAP contributions to explain potential stock market crashes at a given date, while Gradojevic et al. [15] used SHAP to get an insight into option pricing before and during the COVID-19 pandemic.

Recent work proposes and evaluates different explanation styles [16], represented via the theory of Pierce [17], which defines three logical reasoning styles: (i) Inductive, which involves drawing a general conclusion from a set of specific observations; (ii) Abductive, which begins with an incomplete set of observations and proceeds to the likeliest possible explanation, and (iii) Deductive reasoning, which starts with general rules and examines the possibilities to reach a specific, logical conclusion. Results with human decision-makers show that specific explanation styles (abductive and deductive) *improve the user's* ***task performance*** *in the case of high AI confidence* compared to inductive explanations. The explanations proposed in this paper can be seen as abductive, as they describe the suitability of a candidate based on (incomplete) observations of candidate features contributing to a suitability prediction.

In this paper, career path prediction is performed on a dataset provided by Randstad NV[1]. As the world's largest employment agency [18], Randstad has an enormous dataset containing the careers of hundreds of thousands of individuals.

This paper attempts to answer the following research question: *To what degree can career path predictions done by deep learning models be made explainable?* This is done by means of the following sub-questions:

- **RQ1:** How well do state-of-the-art deep learning models perform career path prediction on the dataset?
- **RQ2:** Do different ways of making model predictions explainable impact prediction accuracy?
- **RQ3:** Which explainable model is the most useful for recommending jobs to candidates?

The explanations generated by the models in this paper are evaluated in a functionality-grounded evaluation of explanation fidelity [19]. In this type of evaluation, some formal definition of interpretability serves as a proxy to evaluate the explanation quality. As with previous evaluation frameworks for XAI (c.f.,) [20]), we used human (expert) judgments as the gold standard to evaluate a proxy representation of the model (which features were mentioned). We primarily focus on soundness, e.g., the explanation does not indicate features that are not important. This is different from e.g., the criterion of completeness which aims to represent the entire dynamics of the model, e.g., including all the (important) features in an explanation. We note that while the features are validated in a human-grounded way, this is not a typical human-grounded evaluation where we would validate the explanations in a setting closer to the final application setting. For a further comparison of evaluation methodologies we refer to the surveys by [19] and [21].

The paper is structured as follows: first, an overview of the current state

[1] https://www.randstad.nl/

of the art in terms of model prediction performance and explainability is given (Section 2). Then, Randstad's dataset is described in detail (Section 3). Afterwards, the methods used to answer the research questions are (Section 4). Subsequently, the research questions are answered (Section 5), after which their answers are discussed (Section 6).

2 Related Work

2.1 Career path predictions

The goal of career path prediction is to determine what position of employment is a logical next step given a job seeker's career [5]. Considering the number of different career opportunities and factors which have an influence on the career steps (e.g., previous job experiences, educational background, interests of a job seeker), the career prediction problem is incredibly difficult to model by hand.

In recent years a lot of progress has been achieved within the field of career path prediction. The first notable paper to use machine learning for career path prediction, was that by Liu et al. [5]. In this paper, Liu et al. scraped individuals' social media profiles to generate a dataset, after which they predict when an employee would be ready to move to a higher-paying position within their current field (e.g. moving from junior software developer to senior software developer). Meng et al. [4] then extended this task by not just considering within-field switches, but general job mobility. Their custom LSTM, the *hierarchical career-path-aware neural network* (HCPNN), was thus tasked to predict individuals' next employer, regardless of their current field of employment. The HCPNN has shown impressive results, outperforming every model that forewent it.

Similarly, He et al. [7] attempted to predict individuals' next job based on features they extracted from their resume. Unlike Meng et al., they made use of a convolutional neural network (CNN) for the predictions. With this CNN they tried to implement a multi-purpose model that could not only predict talents' next job position, but also their salary and the size of the company they would be working at. Out of those three tasks, their CNN proved to perform the best on career path predictions.

At their core, Meng et al.'s LSTM and He et al.'s CNN are simply feature extractors that feed their output into a dense layer. While both perform well on their own, it is common to combine these two architectures within the field of sequence classification [22–24]. Although such an architecture has not yet been used for career path predictions specifically, they have been shown to perform exceedingly well on other multivariate sequence classification problems [25–27]. Especially Livieris et al. [26] their CNN-LSTM has shown good results on another multivariate sequence classification task (gold price forecasting), outperforming every alternative architecture tested.

While the aforementioned models make up the current state of the art for career path predictions, they all share a common flaw: they function

as black boxes. As a result, their outputs are hard to interpret for both recruiters and job seekers. Considering the impact a career change can have on an individual's life, this can make the models difficult to use in real-world scenarios.

2.2 Explainability in deep learning

Explainability and model performance (e.g., accuracy, precision, recall) are often considered inverses of each other in the field of AI. A simple, easy to explain model is likely to perform mediocre at best, while a complex, difficult to explain model is more likely to perform well [28]. A common example of this inverse relationship can be seen in the difference between decision trees and random forests: random forests are based on decision trees, but with a higher degree of complexity, which strongly increases performance at the cost of explainability.

However, with the increasing interest in explainable AI, more and more solutions have been brought up that can make even the most complex deep learning models explainable to a degree [29]. Most commonly, this explainability takes the shape of visualizations of the networks' behaviour. Saliency maps and attention distributions are capable of visualizing the importance of different variables, usually through some type of colour scheme indicating higher or lower feature importance. Initially, Springenberg et al. [30] used guided backpropagation to visualize the features learned by convolutional layers. Extending past guided backpropagation, Selvaraju et al. [31] created Grad-CAM, which could not only visualize *general* learned features, but also determine which features were important for a *specific* predicted class. Since these post-hoc interpretability techniques merely look at the behaviour of the model, they do not alter their performance. However, it is often necessary to make alterations to the models' architecture to allow good explanations to be generated (e.g., they only work on convolutional layers, and preferably only on the *final* convolutional layer of a model) [30, 31]. As a result, such techniques either do not change performance at all, or decrease it slightly. In contrast, while both aforementioned methods were created for computer vision, Vaswani et al. [32] proposed 'attention mechanisms' for natural language processing. These attention mechanisms cause the models to predict the importance of each feature per time step (or the importance of a given time step in general) which can then be visualized. As a result, Vaswani et al. made it possible for different model architectures to become explainable, while simultaneously *improving* their performance.

2.3 Explainability in sequence classification

Sequence classification brings an additional factor into the mix: the temporal dimension. Simply visualizing which features garner the most attention thus becomes insufficient in this scenario. While a given variable might be highly important to the network initially, it could become less relevant as time progresses. Thus, to make explainable sequence classifications, not only should there

be an explanation of which variables contributed the most to the final prediction, but also at what moment their values were most decisive [33]. Nonetheless, saliency maps are still useful in this scenario, as a multivariate sequence can be treated as a 2-dimensional image of shape ($Number\ of\ features \times sequence\ length$). However, these saliency maps do not necessarily reach the level of finesse required to generate understandable explanations for sequences. As a result, saliency maps are often combined with attention mechanisms. By combining saliency maps with attention, it is possible to improve the quality of the explanations [34]. Additionally, two distinct attention mechanisms can be used simultaneously - one for the temporal dimension, and one for the spatial (feature-based) aspect. By applying spatial attention at each time step, and then combining that with temporal attention over the time series as a whole, the relative and overall importance of each feature at each time step can be determined [35].

2.4 Evaluating explanations

Although the field of explainable AI has seen rapid growth in recent years, there has yet to be a commonly agreed-upon method for evaluating explanations [36, 37]. Depending on the type of research and the available resources, researchers have opted for a number of different evaluation strategies; most notably: anecdotal evaluation, quantitative evaluation, and user-centered evaluation [38].

Anecdotal evaluation, as the name implies, often relies on the use of anecdotal evidence to support the claim that a model can generate proper explanations. This usually takes the shape of a single explanation generated by the model being presented, accompanied by a discussion of its aptitude. Although this gives a general sense of the type of explanations being generated by the model, it is far from a comprehensive proof of their quality, as the example could be cherry-picked or coincidentally good. As a result, the exclusive use of anecdotal evidence in XAI research has been declining over the year, in favor of more robust evaluations [38].

Quantitative evaluation takes a more mathematical approach, attempting to 'objectively' evaluate the explanations generated by the model. These types of evaluations can take a number of different forms, such as tests for explanation stability, or comparisons to a white box model [39, 40]. Such evaluation methods offer valuable insights into the quality of the explanations, providing researchers with measurable proof. However, such proof is often not the end-all-be-all, as quantitatively proper explanations can still be incomprehensible to users, and quantitatively poor explanations can still be useful to users [41].

Lastly, there are user studies, which make use of (end) users for the evaluation of the explanations [42]. Depending on the concept to be evaluated, these user studies can take different shapes; generally, however, they include the users interacting with the explanations and being

asked to perform different tasks and answer different questions [43]. Considering anecdotal and quantitative evaluations lack user-centered evaluation, they fail to acknowledge the ultimate goal of generating explanations: improving the usability of models for their users. Therefore, user studies are generally regarded as the most valuable method of evaluation. Nevertheless, it is also the least commonly used method of evaluation, mostly due to it being costly to execute, especially when the intended end users of a system should be experts within a given field [38].

2.5 Contributions

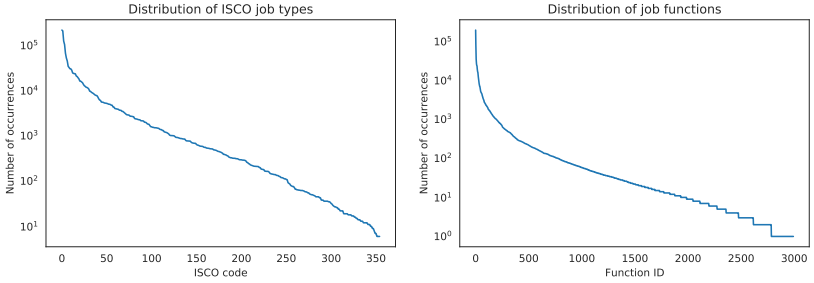
One common limitation in the field of career path prediction is the lack of a high-quality, publicly-accessible dataset. As a result, previous research is distributed amongst many highly disparate datasets, making comparing between models difficult. In this paper, we compare a number of supposedly state-of-the-art career path prediction models on a single, high-quality dataset, which allows for direct apples-to-apples comparisons. As a result, a more clear state of the art can be determined, which will allow future work to improve upon it further without first having to redetermine the current SOTA.

Furthermore, our work is the first to apply XAI techniques to the field of career path prediction. Considering the fact that recruitment is classified as a high-risk field under the proposed EU Artificial Intelligence act [44], combined with the GDPR providing citizens with the right to explanation of algorithmic decisions made about them [45], explainability is likely to become a mainstay within AI for recruitment soon. Therefore, looking into the feasibility of explainable models, as well as evaluating explanations generated using different techniques with real-world users gives valuable insights into how future research can abide by these laws.

Lastly, the explanations generated by the explainable models used in this work were previously only evaluated using quantitative methods [46] or anecdotal evidence [34, 35]. By conducting domain experts (recruiters) for our user study, we make use of *human-grounded* evaluation [47], albeit with experts rather than lay users. While user studies are already rare within the field of XAI (being used in less than a quarter of XAI research), user studies with domain experts are even more uncommon [38]. Therefore, by not only pioneering explainability within the field of career path prediction, but also evaluating it in this manner, we believe our work lays a strong foundation for the future of XAI within the field.

3 Description of the Data

The data on which the models were trained, configured, and tested, was provided by Randstad NV (Randstad). Due to the nature of Randstad's operations, they have an exhaustive data lake consisting of temporal employee-related data. Unfortunately, this dataset is not publicly available due to the large amount of personal data contained within it. Despite our best efforts to ensure transparency and rigor in our approach, the limited accessibility of



(a) Distribution of ISCO job types (b) Distribution of job functions

Fig. 1: The distributions of ISCO job types and job functions (**N = 1664565**). Both use a logarithmic scale for the y-axis.

the dataset could constrain the ability of future research to verify our findings and conduct further investigations.

3.1 Overview of the datasets

Randstad's full dataset consists of over two million jobs relating to more than 500 thousand individuals. These jobs span over multiple decades, going back as far as the early twentieth century. Although Randstad is a multinational company, the used dataset only contains data pertaining to candidates living in the Netherlands. For each job, the dataset includes a number of relevant features, such as the company for which the person worked, the period within which they worked, ISCO[2] classifications of the job, and the specific function that was performed. Both job function and ISCO type can be considered as indicators for the candidate's current career step (i.e., the predicted variable), however, the former is more granular as it takes over 3000 unique values, while the latter takes a mere 355.

Additionally, Randstad stores structured and unstructured profile-specific data, which can be used to describe the profile of a candidate. The structured data includes:

- education history, which includes education level, completion status, the start and (if applicable) end date;
- skills (e.g. 'programming: Python', 'operating a forklift', 'Microsoft Word', etc.);
- languages;
- driving licenses;
- location.

The unstructured data is represented by curriculum vitaes (CVs), which are user-generated documents.

3.2 Data imbalance

Within the dataset, the majority (i.e., 57%) of career steps consist of people merely switching companies, while keeping the same job function. Therefore, predicting that a candidate always stays within their job

[2] https://www.ilo.org/public/english/bureau/stat/isco/isco08/

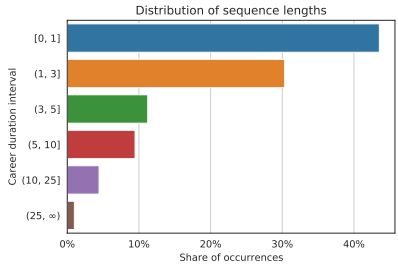


Fig. 2: Distribution of different bins of total number of jobs held by each candidate ($\mathbf{N = 472647}$). Individuals in the $[0, 1]$ bin were removed from the dataset. For the full distribution, see Figure 6.

function forms an unbeatable baseline. Combined with the fact that such career steps are less interesting for a candidate's eventual career path, the dataset was filtered to only contain candidates who made a career *switch* as their last career step.

Furthermore, there is a huge imbalance in work experience and education levels of candidates present in the data. The imbalance in work experience occurs in job positions, which are represented by ISCO job types and job functions (see Figure 1a and 1b respectively), and the number of positions candidates have had (see Figure 2). We addressed the skew in the number of jobs a candidate had by limiting the job history to the 25 most recent jobs. The imbalance in education levels (see Figure 3) is less impactful, as the education level of candidates is merely a predictor, unlike the ISCO job types and job functions, both of which could be used as the actual labels to be predicted. To construct the final dataset we

1. limited the job history of candidates to the 25 most recent jobs;
2. dropped candidates with fewer than two jobs in the dataset, due to the inability to convert their careers to a sequence;
3. balanced class labels distribution through weighted sampling during training.[3]

This resulted in our final dataset consisting of the careers of 113724 candidates, each being limited to the 25 most recent jobs they had. For each job, the (normalized[4]) time spent working there, the ISCO function level of the job, the highest education enjoyed up until then, the company for which the candidate worked, the specific job function ID, the ISCO job type, and the most recent CV were stored. Additionally, the zip code, obtained certificates, mastered languages, skills, and driving licenses of candidates were stored as static variables, since they rarely changed in between jobs.

4 Methodology

In order to make career path predictions, candidates' profiles were turned into sequences which could be fed into different (deep learning) models. For each candidate we used the last 25 jobs along with profile-specific features as input for the models, after which the models would predict their next job in the form of its ISCO

[3] This was done through PyTorch's WeightedRandomSampler, which randomly selects samples from the training set according to a weighted distribution. Samples with a more common label were undersampled, and samples with a less common label were oversampled.

[4] Normalization was done through Z-transformation in order to maintain a common scale for all features.

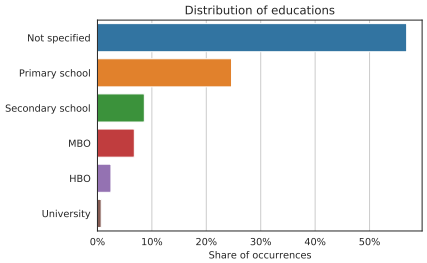


Fig. 3: Distribution of highest education level obtained by candidates (**N = 1664565**). In the Dutch education system, MBO refers to intermediate vocational training, and HBO refers to university of applied sciences.

job type. Candidate profiles that consisted of fewer than 25 jobs were zero-padded to prevent mismatched sequence lengths. This section outlines how candidates' careers were converted into sequences, as well as how those sequences were fed into different models.

Lastly, an overview of the models used is given. The used models can be split into three separate categories: non-neural baselines, non-explainable neural models[5], and explainable neural models. 80% of the candidates were used as a training set, 10% of the candidates were used as a validation set, on which the optimal hyperparameters were determined, and the last 10% of the candidates were used as a test set to evaluate model performance on unseen data. We used weighted sampling during training to address the imbalance within the class labels distribution.

[5]The neural models were created in PyTorch and trained on an NVIDIA tesla K80 GPU [48].

4.1 Data Preprocessing

Due to the availability of temporal data, candidates' career paths were turned into sequences. For these sequences, each job held by a candidate was considered to be one time step. The order of the time steps was determined by the date at which the candidate started the position. As a result, every career was turned into a sequence, in which each time step was a candidate's current job, combined with their location and the skills, certificates, languages, and education they had achieved at the time of starting the position. Each feature was one- or multi-hot encoded and converted to embedding vectors during training. To also include candidates' curriculum vitaes (CVs) at each time step, the most recent CV uploaded by a candidate at each time step was converted to numerical features using averaged Word2Vec [49] embeddings and combined with the other features. Although Word2Vec was used due to it already being commonplace at Randstad, embeddings generated by more sophisticated methods could be used by future work to study the effect of the CV embeddings on prediction accuracy. The embedding sizes of all features can be found in Appendix B.

4.2 Baselines and Models

Considering the fact that careers do not necessarily follow a logical trend, they can be rather difficult to model properly [4]. To evaluate the added value of using deep learning models, and to allow for better contextualization, baselines were set with three

non-deep learning (but coincidentally highly explainable) models.

The first one is a simple majority class baseline, which always predicts the most common job in the dataset.

The second baseline is the majority *switch*, which always predicts the most common job following the current job of the candidate. Considering all same-job switches were removed from the dataset, this strategy always predicts an entirely new job type.

The last simple baseline is more sophisticated: k-nearest neighbors based on the dynamic time warping distance between candidates that had the same previous job (KNN-DTW). This baseline uses dynamic time warping [50] to line up different candidates' careers, after which it determines the distance between the two. Based on this distance the 'career similarity' of two candidates can be calculated, after which k-nearest neighbors can be used to make a prediction. For each candidate, only candidates that had the same previous job were compared in terms of DTW distance to simultaneously improve performance and decrease computation costs.

4.2.1 RQ1 - State of the art

To study the impact of explainability mechanisms on model performance, three state-of-the-art (SOTA) models, each with a unique architecture (Section 2.1), were trained and tested on Randstad's dataset. The performance of these models will function as a non-explainable baseline, with which the performance of the explainable alternatives can be compared. All models, both SOTA and explainable, were trained to perform the exact same task and therefore have the same output layer: a linear layer with 355 neurons (one for each ISCO code present in the dataset) followed by a softmax function. This allows the models to output a probability distribution over the possible classes, where the class with the highest probability is considered to be the actual prediction. The following SOTA models were used:

LSTM: The LSTM-based model used in this paper is based on the HCPNN by Meng et al. [4]. While the original HCPNN combines candidate-specific data with company-specific data, its modular architecture allows for the removal of some of the model's components. As a result, the HCPNN was implemented using only candidate-specific features. This results in a model that takes embedded position features, feeds them into an LSTM, runs the LSTM's output through an attention layer, and combines that output with a candidate's embedded static features. Lastly, the aforementioned output layer is responsible for the final prediction.

CNN: The CNN-based model used in this paper is that of He et al. [7]. This architecture feeds the input data into a 2D convolutional layer, followed by a pooling layer. The output is then flattened and ran through a drop-out layer. Lastly, the output layer is used to make the final prediction.

CNN-LSTM: The CNN-LSTM-based model used in this paper is based on the model created

by Livieris et al. [26]. It uses two sequential 2D convolutional layers, followed by a pooling layer. The pooled features then get fed into an LSTM, after which the output layer gives the final prediction of the model.

To evaluate performance, accuracy @ k ($k \in \{1, 5, 10\}$) was used, which shows how often the correct answer was within the top k predictions given by the model [51]. Considering the fact that candidates could not be interested in a specific job type (e.g. no open vacancies, not interesting enough, it pays too little), it is expected of recruiters that they can provide multiple recommendations for the candidate, allowing them to choose and consider multiple options.

4.2.2 RQ2 - Explainable models

Although the explainable models' architectures differ slightly from the aforementioned state-of-the-art models to allow for improved explainability, they are largely identical.

Explainable LSTM : The explainable LSTM-based model (eLSTM) used in this paper is based on the *spatiotemporal attention LSTM* (STA-LSTM) by Ding et al. [35]. This architecture starts off by determining spatial attention; it runs each individual time step through a linear layer, after which the Hadamard product between the linear layer's output and the features per time step is taken to determine the importance of each feature at each time step. The output hereof is then fed into an LSTM, after which the temporal attention is calculated. This is done by flattening the output of the LSTM and running it through another linear layer. This calculates a normalized importance of each time step, based on that step's hidden values. The dot product between the linear layer's output and the LSTM's hidden output is then calculated, which is fed into the output layer to make the final predictions.

Explainable CNN : The explainable CNN-based model (eCNN) used in this paper is based on the *explainable convolutional neural network for multivariate time series classification* (XCM) by Fauvel et al. [46]. It makes use of two stages that run in parallel. The first stage (top) uses two sequential 2D convolutional layers to process the feature values, while the other stage (bottom) uses two *1D* convolutional layers to independently process the temporal dimension. The outputs of both stages are then concatenated, after which another 1D convolutional layer generates a given number of feature maps (the specific number is a hyperparameter). These feature maps are then fed into a pooling layer, after which the output layer makes the prediction.

Explainable CNN-LSTM : The explainable CNN-LSTM-based model (eCNN-LSTM) used in this paper is based on that of Schockaert et al. [34]. This model runs the input data through a 2D convolutional

layer whose output gets concatenated to the original sequential data. This combined output gets fed into an LSTM. All but the last hidden state of the LSTM get passed through a temporal attention mechanism. This temporal attention mechanism runs each hidden state through a fully-connected layer which attributes it a given amount of attention. These attention values are then normalized, after which the dot product of the attention vector and the hidden states is calculated to create a *context vector*. This context vector is then concatenated to the last hidden state of the LSTM, and fed into the output layer, which makes the final prediction.

Though the model architectures differ widely, they are all able to generate the same three types of explanations: spatial, temporal, and spatiotemporal.

4.2.3 RQ3 - Real-world utility

A user study was conducted to measure the adequacy of the explanations generated by the models. Potential users of the models (e.g. Randstad's recruiters), were tasked to determine which variables were most relevant for a prediction made by the system. Six recruiters were split into three groups based on their recruiting expertise (finance, customer support, health care), and shown three separate predictions within that industry (one per model). For each prediction, they were tasked to distribute 100 'relevance points' over all of the features used by the models (previous jobs, education, skills, etc.), after which their distribution was compared to that of the models. In order to determine the similarity between the models' explanations and those generated by recruiters, the Pearson's correlation, root mean squared error (RMSE), and mean absolute error (MAE) of each models' distributions compared to the recruiters' distributions were calculated. I.e., by doing so, we evaluate the *coherence* of the explanations according to the recruiters - the plausibility of the explanation according to humans [38].

Furthermore, the recruiters were presented with the explanations generated by each model, and tasked to judge each part of the explanations (spatial/feature attention, temporal attention, and spatiotemporal attention), as well as the general usefulness of the explanations for finding a suitable position for a candidate. Thus, in addition to their distributions, each recruiter also provided 4 evaluations per model, for a total of 12 ratings per recruiter. By averaging the scores given by the recruiters, the *context* of the explanations is determined, i.e., the extent to which the end users' needs are taken into account by the explanations [38].

5 Results

5.1 RQ1 - State of the art

To better convey the performance gained by using deep learning models, the score of each model will be directly compared to that of the best-performing baseline. Of the three simple baselines, the majority switch

baseline performed the best, reaching 19.1% accuracy @ 1, 46.6% accuracy @ 5, and 61.3% accuracy @ 10. KNN-DTW performed worse initially, but converged to the majority switch baseline as the number of neighbors (K) approached infinity. With low values of K, e.g. 5, it failed to break even 10% accuracy @ 1. However, using a higher value for K, e.g. 100, greatly improved this score, reaching 18.1% accuracy @ 1, 46.4% accuracy @ 5, and 58.1% accuracy @ 10, showing a sub-linear performance gain as K increased. The majority class baseline performed significantly worse, only reaching 10.5% accuracy @ 1, 36.8% accuracy @ 5, and 49.1% accuracy @ 10. As a result, the performance of the deep learning models was compared against the scores achieved by the majority switch baseline.

While similar architectures were used for the explainable and non-explainable models, different hyperparameter configurations led to different accuracy values for each architecture. The results shown in Table 1 only indicate the performance given by the best hyperparameter configuration found for each model. For a full overview of hyperparameter configurations and their related performance see Appendix C.

5.2 RQ2 - Explainable models

Out of all the models, the CNN-LSTMs performed the best. Unlike what was hypothesized, the explainable models were not inferior to their non-explainable counterparts (Table 1). In fact, the eLSTM provides a higher accuracy than the non-explainable LSTM by a slight margin, although this difference falls within the confidence intervals of the scores, and is therefore not significant ($p > .05$). The explainable CNN took a slight (but statistically significant) hit in performance in exchange for the increase in explainability, especially suffering at higher values of k.

5.3 RQ3 - Real-world utility

Each explainable model is able to generate three separate explanations for a prediction: (i) the weight of each feature, (ii) the weight of each time step, and (iii) a time step/feature interaction map (spatiotemporal attention). The way in which these explanations are generated differs per model, but the final visualizations are the same, regardless of the method used to generate them (Figure 10, 11, and 12 in Appendix F).

In order to verify the validity of these explanations, user research was done with Randstad's recruiters. After providing the recruiters with the predictions made by the model,

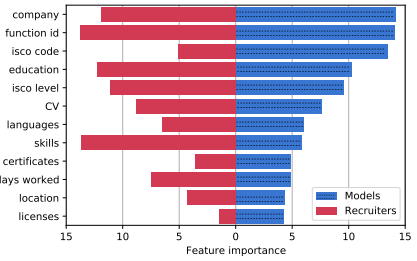


Fig. 4: Average distribution of feature importance of the three explainable models compared to that of Randstad's recruiters (**N = 18**).

Model	Accuracy @ 1 ⇑	Accuracy @ 5 ⇑	Accuracy @ 10 ⇑
	Baseline		
Majority Switch	19.1% ± 0.7%	46.6% ± 0.9%	61.3% ± 0.9%
	Non-explainable models		
LSTM	21.9% ± 0.8%**	49.3% ± 0.9%*	62.9% ± 0.9%
CNN	20.8% ± 0.7%*	50.8% ± 0.9%***	63.7% ± 0.9%*
CNN-LSTM	26.4% ± 0.6%***	56.5% ± 0.7%***	68.6% ± 0.6%***
	Explainable models		
eLSTM	22.2% ± 0.8%*	47.6% ± 0.9%	60.8% ± 0.9%
eCNN	20.1% ± 0.7%	47.7% ± 0.9%	61.5% ± 0.9%
eCNN-LSTM	26.0% ± 0.8%***	55.7% ± 0.9%***	67.5% ± 0.9%***

Table 1: Test set performance of each model at different values of k ($\mathbf{N = 11372}$). Different values of k indicate how often the correct answer was within the top k predictions given by the model. Asterisks indicate significant differences compared to the majority switch baseline; $^*p < 0.05$, $^{**}p < 0.01$, $^{***}p < 0.001$.

they were asked to estimate which variables were most important. The averaged estimates made by the recruiters and models can be seen in Figure 4 (for the comparison per model see Appendix D). The results indicate that the models' explanations were positively correlated with those made by the recruiters (Table 2). For the eCNN-LSTM, this correlation was moderate, while for the eCNN and eLSTM, it was quite weak. Interestingly, the models and recruiters largely agreed on which features were most relevant, but disagreed on the importance of skills and previous jobs (in terms of ISCO codes). While the models determined skills to be one of the least important features, recruiters actually found them to be the single most valuable feature. Inversely, the models attributed a lot of weight to previous ISCO codes, while recruiters found them mostly irrelevant. This could indicate that recruiters are more willing to take 'risks' when providing candidates with recommendations, by putting more emphasis on a candidate's abilities, even when they may lack relevant experience in the field.

To measure the similarity between the explanations generated by the models and recruiters, three metrics were calculated for each of them: RMSE, MAE, and Pearson correlation. This was done by calculating the difference between the average score that recruiters gave to each feature and the attention put towards that feature by the models (RMSE and MAE), as well as the correlation between the models' values and the recruiters' values (Pearson correlation). The results can be seen in Table 2.

Additionally, the recruiters were asked how understandable they found the models' explanations, as well as how useful they considered the models (including their explanations) for

	Pearson's r ⇑	**RMSE ⇓**	**MAE ⇓**
eLSTM	0.142	4.661	4.094
eCNN-LSTM	0.436	6.014	4.847
eCNN	0.152	5.594	4.518

Table 2: The Pearson correlation, RMSE, and MAE of each model compared to the scores given by the recruiters (**N = 6**). For each feature, both the models and the recruiters gave a score; the scores are calculated based on those two scores.

helping candidates find a new job. The averaged scores for each model is shown in Table 3.

In general, the recruiters showed a preference for the feature explanations, and to a lesser extent the spatiotemporal explanations. The temporal explanations were considered the least sensible, failing to reach a sufficient grade (i.e., above a 5.5/10 on average). While the eCNN was judged to deliver the worst explanations, receiving barely a 5/10 on average, the eCNN-LSTM's and eLSTM's explanations were considered sufficient by the recruiters. Out of these two, the eCNN-LSTM was determined to provide the best explanations, scoring the highest average rating in each category. Regardless of the insufficient grades reached by some explanations/models, all three models were considered generally useful for recommending a job to a candidate.

6 Discussion and conclusion

6.1 Interpretation of the results

6.1.1 State of the art performance

Although career path prediction is a notoriously difficult problem in deep learning, the state-of-the-art models used on Randstad's dataset ended up performing commendably. All three models ended up achieving significantly ($p < .05$) higher scores than the majority switch baseline, which already performed well. However, this improvement is relatively small for the CNN and LSTM. This marginal increase over the baseline is largely in line with the results found in previous research. Meng et al. [4] found that the HCPNN outperformed non-neural baselines by about 20% on their dataset; improving from 6.0%

	Feature explanation	**Temporal explanation**	**Spatiotemporal explanation**	**General usability**
eLSTM	6.4 (SD=2.30)	5.4 (SD=2.30)	5.4 (SD=1.14)	6.0 (SD=0.71)
eCNN	5.2 (SD=1.79)	4.6 (SD=2.70)	5.4 (SD=2.07)	6.4 (SD=1.14)
eCNN-LSTM	6.6 (SD=2.51)	5.4 (SD=1.67)	6.4 (SD=2.41)	6.8 (SD=1.10)

Table 3: The average rating of each type of explanation for each model, as well as their general usability score, as determined by Randstad's recruiters (**N = 5**). 1-10 scale.

to 7.3% accuracy @ 1. Although this is a larger improvement than that of the HCPNN compared to the majority switch baseline presented in this paper (14.6% increase in accuracy @ 1), this result can still be considered a confirmation of Meng et al. their findings. The smaller relative improvement could in part be caused by the fact that Randstad's dataset includes data that has been manually input by candidates themselves. This data, as opposed to that input by Randstad's recruiters, has not been verified, and could therefore include errors, a substantial amount of missing values, etc. While these data points could have been removed from the dataset to improve performance, a conscious decision was made not to. Removing all data entered by candidates themselves would get rid of more than half the dataset, in exchange for a relatively minor improvement in performance (in the neighborhood of 5-10%, absolute). Additionally, in real-world use, providing candidates with the ability to enter their own career into Randstad's system and instantly being able to receive job recommendations is very valuable.

As opposed to the CNN and LSTM, the CNN-LSTM showed a major improvement over the baseline. This is in accordance with the results found by Livieris et al. [26], who showed that their CNN-LSTM significantly outperformed a bare LSTM baseline. Considering the fact that both the convolutional layers and LSTM layers are used as feature extractors, this result is expected. By combining the two layer types, the model is able to learn more abstract representations of the data, allowing it to generalize better [52–54].

6.1.2 Explainability's impact on performance

Though it was initially expected that the inclusion of explainability mechanisms would impact model performance to a degree [28], the experiments have shown that this is not the case. While for Grad-CAM (CNN) this result might seem obvious, considering this technique does not alter the model, but merely looks at the model's gradients, this is still surprising. Despite the fact that the technique itself is not intrusive, the model's architecture still needed to be altered in order to create functioning explanations (e.g. the eCNN's parallel design), as shown by Fauvel et al. [46]. Regardless of this architectural change, however, the explainable model still performed on-par with its counterpart. Similarly, the explainable CNN-LSTM, which uses not only guided backpropagation, but also an attention mechanism, showed roughly equal performance to the non-explainable CNN-LSTM. For the LSTM, the addition of explainability even improved the model's performance (in terms of accuracy @ 1), although this improvement was not statistically significant. Thus, the experiments show that explainability mechanisms can be used in deep learning models for career path prediction without hindering the models' predictive powers. For the most part, this is in line with the results of previous research on the topic [30, 31]. However, the fact that the attention mechanisms used in the eCNN-LSTM and eLSTM did not

improve model accuracy in a statistically significant manner is in stride with the results found by Schockaert et al. [34] and Ding et al. [35]. This is likely caused by the differences between their datasets and the one provided by Randstad. For example, the majority of candidates in the filtered dataset only had two job on record - one to use for the prediction, and one that provides the ground truth. In such a scenario, temporal attention adds no value, as all attention will be directed towards that single usable time step.

6.1.3 Real-word utility

Evaluation with recruiters found that they consider the explainable models usable in a real-world scenario. Although they were quite critical, giving mostly sufficient (but not outstanding) grades, they determined that each model type would at least be helpful to a degree in finding a job for a candidate. The individual explanation types tended to score lower than the models as a whole, indicating that the current implementation of the models' explanations (i.e. the visualizations in Appendix F) might require some tuning or extra clarification in order to be used efficiently by recruiters. Regardless, the recruiters did indicate that they considered the current implementation useful as is. Considering the environment for the user study is quite barebones (Appendix E), this is a positive indication for the actual usability of the models' explanations.

By improving the clarity of the explanations, the models might also become usable by candidates themselves. Considering the inference time of the models (less than a second), candidates could enter their careers into Randstad's system, and instantly be provided a list of job recommendations, accompanied by explanations. However, more research will need to be done to determine if this is preferable for candidates over having recruiters interpret the models' predictions.

6.2 Potential biases

While the models performed commendably, and the explanations were determined to be satisfactory, it is important to consider the impact of biases in the training data on the predictions. Although protected features, such as gender, race, and age were removed from the dataset, correlation between such features and input features may still have caused discrimination [55]. For example, while age was not explicitly present in the data, the models could still roughly determine a candidate's age based on their total number of days worked across all jobs (a person with a few hundred total days worked is likely to be in their twenties, while someone with over ten thousand days worked is probably nearing retirement). The models' ability to 'retrieve' such protected features may have negatively affected the recommendations for specific candidates. Future research could look into the extent to which this occurs, as well as methods to alleviate this effect.

6.3 Limitations and expansion

Due to the lack of a publicly available dataset, determining state-of-the-art performance is complicated for career path prediction. Even within Randstad's own dataset, performance could be increased by simply filtering out data entered by candidates. To advance the field of career path prediction, future research should focus on creating a general dataset that can be used to directly compare model performance within the field (in the same vein as ImageNet for image classification[6] and TREC for text retrieval[7]). This benchmarking dataset should consist of relatively clean, GDPR compliant, exhaustive career data of a large variety of candidates. Using this dataset, future research will be able to better gauge the performance of different architectures used for career path prediction (e.g. LSTMs, CNNs, temporal graphs) and draw direct comparisons between models. Thus, having a clear and definite state of the art will most certainly advance the field as a whole.

Another limitation posed in this paper, is the lack of hardware resources. The NVIDIA Tesla K80 used to train the models fell short when training the CNN-based models. Because of the low CUDA core count of 2496, and the limited 12 gigabytes of VRAM, the convolutional models had to be limited in terms of kernel size, output channels, embedding sizes, epochs, and batch sizes to decrease VRAM usage and keep training time reasonable. Consequently, not all possible hyperparameter configurations could be tested, possibly leaving better model configurations unexplored.

Furthermore, the small sample size used for the user study is an important limitation to acknowledge. Because the participating recruiters were on payroll, it was difficult to get their managers' approval, as well as to schedule a moment to perform the tests. Subsequently, the results gathered by the user study are subject to high variance and are therefore difficult to use as conclusive evidence. Increasing the sample size by also performing user studies on candidates themselves would have helped solve this issue and might have provided additional insights. Also, improving the clarity of the UI used for the user study and the models' explanations could have led to lower variance, making the results more conclusive.

Additionally, while only including career switches in the training data strongly improved the models' usability, it also hinders individuals who are looking for new work within their current field from receiving recommendations. To account for such candidates, future work could expand upon the current pipeline by including a recommendation on whether a candidate should stay within their current field, or pursue a position with a different function. For individuals who get recommended to stay within their profession, the models could, for example, be altered to recommend a next employer within the field.

[6] https://www.image-net.org/
[7] https://trec.nist.gov/data.html

6.4 Conclusion

In the span of this paper, it was shown that career path predictions made by deep learning models can be made explainable to a high degree. While different types of explanations made by the models can differ in terms of how understandable they are to humans, all of them turned out to be useful for recruiters nonetheless. Due to the fact that these explainability mechanisms do not lead to a decrease in performance, they form a good addition to existing career path prediction models. This goes especially for CNN-LSTMs, as those perform the best as explainable and non-explainable models, while also providing the best explanations according to recruiters.

Data availability statement

The data that support the findings of this study are available from Randstad N.V. However, these data contain highly personal information, and were therefore only available under licence for the current study.

References

[1] Parigi, P., Ma, X.: The gig economy. XRDS: Crossroads, The ACM Magazine for Students **23**(2), 38–41 (2016)

[2] Autor, D., *et al.*: The polarization of job opportunities in the us labor market: Implications for employment and earnings. Center for American Progress and The Hamilton Project **6**, 11–19 (2010)

[3] Zimmermann, T., Kotschenreuther, L., Schmidt, K.: Data-driven hr-résumé analysis based on natural language processing and machine learning. arXiv preprint arXiv:1606.05611 (2016)

[4] Meng, Q., Zhu, H., Xiao, K., Zhang, L., Xiong, H.: A hierarchical career-path-aware neural network for job mobility prediction. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 14–24 (2019)

[5] Liu, Y., Zhang, L., Nie, L., Yan, Y., Rosenblum, D.: Fortune teller: predicting your career path. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30, pp. 1–16 (2016)

[6] Kokkodis, M., Ipeirotis, P.G.: Demand-aware career path recommendations: A reinforcement learning approach. Management Science **67**(7), 4362–4383 (2021)

[7] He, M., Shen, D., Zhu, Y., He, R., Wang, T., Zhang, Z.: Career trajectory prediction based on cnn. In: 2019 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), pp. 22–26 (2019). IEEE

[8] Van Lent, M., Fisher, W., Mancuso, M.: An explainable artificial intelligence system for small-unit tactical behavior. In: Proceedings of the National Conference on Artificial Intelligence, pp. 900–907 (2004). Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999

[9] Nourani, M., Kabir, S., Mohseni, S., Ragan, E.D.: The effects of meaningful and meaningless explanations on trust and perceived system accuracy in intelligent systems. In: Proceedings of the AAAI Conference on Human Computation and Crowdsourcing, vol. 7, pp. 97–105 (2019)

[10] Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)

[11] Lundberg, S.M., Lee, S.-I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)

[12] Bandi, H., Joshi, S., Bhagat, S., Ambawade, D.: Integrated technical and sentiment analysis tool for market index movement prediction, comprehensible using xai. In: 2021 International Conference on Communication Information and Computing Technology (ICCICT), pp. 1–8 (2021). https://doi.org/10.1109/ICCICT50803.2021.9510124

[13] Gite, S., Khatavkar, H., Kotecha, K., Srivastava, S., Maheshwari, P., Pandey, N.: Explainable stock prices prediction from financial news articles using sentiment analysis. PeerJ Computer Science **7**, 340 (2021). https://doi.org/10.7717/peerj-cs.340

[14] Benhamou, E., Ohana, J.-J., Saltiel, D., Guez, B.: Explainable AI (XAI) models applied to planning in financial markets. SSRN Electronic Journal (2021). https://doi.org/10.2139/ssrn.3862437

[15] Gradojevic, N., Kukolj, D.: Unlocking the black box: Non-parametric option pricing before and during covid-19. Annals of Operations Research (2022). https://doi.org/10.1007/s10479-022-04578-7

[16] Cau, F.M., Hauptmann, H., Spano, L.D., Tintarev, N.: Supporting high-uncertainty decisions through ai and logic-style explanations. In: IUI (to Appear) (2023)

[17] Flach, P., Kakas, A.: Abductive and inductive reasoning: Background and issues (2000). https://doi.org/10.1007/978-94-017-0606-3-1

[18] Department, S.R.: Staffing industry: Leading companies worldwide (2022). https://www.statista.com/statistics/257876/staffing-companies-worldwide-by-revenue/

[19] Zhou, J., Gandomi, A.H., Chen, F., Holzinger, A.: Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics **10**(5), 593 (2021)

[20] DeYoung, J., Jain, S., Rajani, N.F., Lehman, E., Xiong, C., Socher, R., Wallace, B.C.: ERASER: A benchmark to evaluate rationalized NLP models. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 4443–4458. Association for Computational Linguistics, Online (2020). https://doi.org/10.18653/v1/2020.acl-main.408. https://aclanthology.org/2020.acl-main.408

[21] Nauta, M., Trienes, J., Pathak, S., Nguyen, E., Peters, M., Schmitt, Y., Schlötterer, J., van Keulen, M., Seifert, C.: From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. ACM Comput. Surv. (2023). https://doi.org/10.1145/3583558. Just Accepted

[22] Vidal, A., Kristjanpoller, W.: Gold volatility prediction using a cnn-lstm approach. Expert Systems with Applications **157**, 113481 (2020)

[23] Lu, W., Li, J., Li, Y., Sun, A., Wang, J.: A cnn-lstm-based model to forecast stock prices. Complexity **2020** (2020)

[24] Rick, R., Berton, L.: Energy forecasting model based on cnn-lstm-ae for many time series with unequal lengths. Engineering Applications of Artificial Intelligence **113**, 104998 (2022)

[25] Kim, T.-Y., Cho, S.-B.: Predicting residential energy consumption using cnn-lstm neural networks. Energy **182**, 72–81 (2019)

[26] Livieris, I.E., Pintelas, E., Pintelas, P.: A cnn–lstm model for gold price time-series forecasting. Neural computing and applications **32**(23), 17351–17360 (2020)

[27] Xie, H., Zhang, L., Lim, C.P.: Evolving cnn-lstm models for time series prediction using enhanced grey wolf optimizer. IEEE Access **8**, 161519–161541 (2020)

[28] Gunning, D., Stefik, M., Choi, J., Miller, T., Stumpf, S., Yang, G.-Z.: Xai—explainable artificial intelligence. Science robotics **4**(37), 7120 (2019)

[29] Choo, J., Liu, S.: Visual analytics for explainable deep learning. IEEE computer graphics and applications **38**(4), 84–92 (2018)

[30] Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806 (2014)

[31] Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual

explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017)

[32] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)

[33] Rojat, T., Puget, R., Filliat, D., Del Ser, J., Gelin, R., Díaz-Rodríguez, N.: Explainable artificial intelligence (xai) on timeseries data: A survey. arXiv preprint arXiv:2104.00950 (2021)

[34] Schockaert, C., Leperlier, R., Moawad, A.: Attention mechanism for multivariate time series recurrent model interpretability applied to the ironmaking industry. arXiv preprint arXiv:2007.12617 (2020)

[35] Ding, Y., Zhu, Y., Feng, J., Zhang, P., Cheng, Z.: Interpretable spatio-temporal attention lstm model for flood forecasting. Neurocomputing **403**, 348–359 (2020)

[36] Markus, A.F., Kors, J.A., Rijnbeek, P.R.: The role of explainability in creating trustworthy artificial intelligence for health care: a comprehensive survey of the terminology, design choices, and evaluation strategies. Journal of Biomedical Informatics **113**, 103655 (2021)

[37] Chromik, M., Schuessler, M.: A taxonomy for human subject evaluation of black-box explanations in xai. Exss-atec@ iui **1** (2020)

[38] Nauta, M., Trienes, J., Pathak, S., Nguyen, E., Peters, M., Schmitt, Y., Schlötterer, J., van Keulen, M., Seifert, C.: From anecdotal evidence to quantitative evaluation methods: A systematic review on evaluating explainable ai. arXiv preprint arXiv:2201.08164 (2022)

[39] Hariharan, S., Rejimol Robinson, R., Prasad, R.R., Thomas, C., Balakrishnan, N.: Xai for intrusion detection system: comparing explanations based on global and local scope. Journal of Computer Virology and Hacking Techniques, 1–23 (2022)

[40] Amparore, E., Perotti, A., Bajardi, P.: To trust or not to trust an explanation: using leaf to evaluate local linear xai methods. PeerJ Computer Science **7**, 479 (2021)

[41] Ye, X., Durrett, G.: The unreliability of explanations in few-shot prompting for textual reasoning. Advances in neural information processing systems (2022)

[42] Kenny, E.M., Ford, C., Quinn, M., Keane, M.T.: Explaining black-box classifiers using post-hoc explanations-by-example: The effect of explanations and

error-rates in xai user studies. Artificial Intelligence **294**, 103459 (2021)

[43] Tintarev, N., Masthoff, J.: Designing and evaluating explanations for recommender systems. In: Recommender Systems Handbook, pp. 479–510. Springer, ??? (2010)

[44] Kop, M.: Eu artificial intelligence act: the european approach to ai. (2021). Stanford-Vienna Transatlantic Technology Law Forum, Transatlantic Antitrust ...

[45] European Commission: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance). European Commission (2016). https://eur-lex.europa.eu/eli/reg/2016/679/oj

[46] Fauvel, K., Lin, T., Masson, V., Fromont, É., Termier, A.: Xcm: An explainable convolutional neural network for multivariate time series classification. Mathematics **9**(23), 3137 (2021)

[47] Doshi-Velez, F., Kim, B.: Considerations for evaluation and generalization in interpretable machine learning. Explainable and interpretable models in computer vision and machine learning, 3–17 (2018)

[48] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)

[49] Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

[50] Berndt, D.J., Clifford, J.: Using dynamic time warping to find patterns in time series. In: KDD Workshop, vol. 10, pp. 359–370 (1994). Seattle, WA, USA:

[51] Powers, D.M.: Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061 (2020)

[52] Eldan, R., Shamir, O.: The power of depth for feedforward neural networks. In: Conference on Learning Theory, pp. 907–940 (2016). PMLR

[53] Subasi, A.: Chapter 5 - other classification examples. In: Subasi, A. (ed.) Practical Machine Learning for Data Analysis Using Python, pp. 323–390. Academic Press, ??? (2020). https://doi.org/10.1016/B978-0-12-821379-7.00005-9.

https://www.sciencedirect.com/science/article/pii/B9780128213797000059

[54] Chen, Y., Zhong, K., Zhang, J., Sun, Q., Zhao, X.: Lstm networks for mobile human activity recognition. In: 2016 International Conference on Artificial Intelligence: Technologies and Applications, pp. 50–53 (2016). Atlantis Press

[55] Träuble, F., Creager, E., Kilbertus, N., Locatello, F., Dittadi, A., Goyal, A., Schölkopf, B., Bauer, S.: On disentangled representations learned from correlated data. In: International Conference on Machine Learning, pp. 10401–10412 (2021). PMLR

[56] Contributors, T.: Torch.sparse¶. PyTorch (2022). https://pytorch.org/docs/stable/sparse.html

[57] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

7 Appendix

All code used in the experiments can be found on https://github.com/Roan-Schellingerhout/MSc_thesis.

A Encoding and indexing

With over 100 thousand careers, each spanning 25 time steps, and over 1000 features per time step (embedding values for skills, certificates, previous jobs, previous companies, addresses, and spoken languages, as well as 300 w2v dimensions per CV), feeding the data into deep learning models as is, turned out to be infeasible. Making use of sparse vectors to lower memory usage also was impossible, due to the incompatibility between CUDA and sparse vectors/matrices [56]. However, considering the large amount of duplicate data (a candidate's skills/certificates/CVs do not change at every time step, and can therefore often be repeated), use was made of indices in order to lower memory usage, at the cost of a slight time complexity increase. For each candidate, a location within each index was created that contained their unique attributes, and the time steps from which those attributes became the most recent ones. By then retrieving the relevant attributes for each candidate in a batch during training, the required memory usage was lowered drastically.

B Embeddings

The embedding size of each feature can be seen in Table 4. Embedding

sizes were determined during hyperparameter tuning.

Feature	Embedding size
CV	300
Company name	300
Function ID	250
ISCO code	150
Skills	100
Certificates	50
Address	25
Languages	15
Education	10
Driver's licenses	10
ISCO level	10

Table 4: The embedding sizes of each feature type.

C Hyperparameters

All hyperparameter tuning results can be found on GitHub. For each configuration, the models were ran for 3 epochs. Based on the results after those 3 epochs, the best performing configuration was ran for 20 epochs to find the optimal number of epochs. Not every intended hyperparameter configuration could be tested due to hardware/time constraints. For example, the CNN-based models needed to be limited to small kernels and output channels to prevent running out of VRAM. Additionally, the eCNN was only trained for a total of 3 epochs, due to time constraints (as each epoch took nearly 8 hours).

All models were optimized using the Adam optimizer [57] (learning rate $= 1 * 10^{-3}$) with cross-entropy loss. The hyperparameters used for the results of the non-explainable models in Table 1 were the following:

LSTM : The HCPNN used a batch size of 512 and reached optimal performance after 18 epochs. It used a single LSTM layer with hidden size 1000.

CNN : The CNN used a batch size of 128 and reached optimal performance after 11 epochs. The 2D convolutional layer consisted of a (5×5) kernel, with (1×1) padding and stride, and generated 64 feature maps. The 3D max-pooling used a $(64 \times 1 \times 1)$ kernel with $(1 \times 1 \times 1)$ stride.

CNN-LSTM : The CNN-LSTM used a batch size of 128 and reached optimal performance after 20 epochs. The first 2D convolutional layer used a (1×1) kernel, with a (1×1) stride and half padding, and generated 32 feature maps. The second 2D convolutional layer made use of the same kernel size, stride, and padding, but generated 64 feature maps. The following 3D average-pooling layer used a $(64 \times 1 \times 1)$ kernel and a $(1 \times 1 \times 1)$ stride. Lastly, the model used a single LSTM layer with hidden size 1000.

The optimal hyperparameters found for the explainable models are as follows:

eLSTM : The explainable LSTM used a batch size of 128 and reached optimal performance after 5 epochs. It used a single LSTM layer with hidden size 1000.

eCNN : The explainable CNN used a batch size of 128 and reached optimal performance after 2 epochs. The top part used a 2D convolutional layer with a (5×1)

kernel (thus, $window\ size = 5$), a (1×1) stride, half padding, and generated 8 feature maps (thus, $F1 = 8$). For the bottom part, the 1D convolutional layer used a $(5 \times N\,features)$ kernel, a (1×1) stride, half padding, and also generated 8 feature maps. The final 1D convolutional layer used a kernel size of $(5 \times (N\,features + 1))$, a (1×1) stride, half padding, and generated 32 feature maps (thus, $F2 = 32$). These 32 feature maps were then ran through an 3D average-pooling layer with kernel size $(32 \times 1 \times 1)$ and a $(1 \times 1 \times 1)$ stride.

eCNN-LSTM : The explainable CNN-LSTM used a batch size of 2048 and reached optimal performance after 15 epochs. Its 2D convolutional layer used a kernel of size $(sequence\,length \times 1)$ and half padding, and was followed by a single LSTM with hidden size 1000.

D Recruiter vs. model distributions

The distributions of feature importance on which Table 2 is based can be seen in Figures 5a, 5b, and 5c. Each model distribution is based on the average feature importance determined by the models across the three categories (finance, health care, and customer support). For the recruiter distribution, the average is taken over the three industries, as well as all recruiters within those industries (as a result, $N = 6$ for all recruiter distributions).

E User study

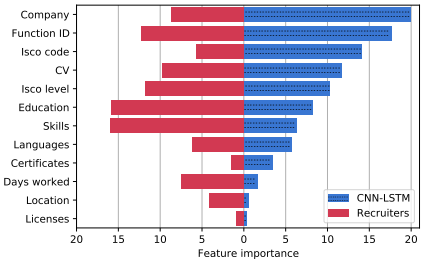
A user study was conducted using a web environment accessible by the recruiters. The web app was hosted using Amazon ec2 in combination with Docker, and built using Flask, JQuery, Jinja, and AJAX. The recruiters were tasked to enter their e-mail address (to allow follow-up questions if needed) and select their expertise (finance, health care, customer support). Afterwards, they were redirected to the first example relevant to their expertise. On this page, the recruiters were shown the data related to the candidate in question (Figure 7), the prediction made by the model, as well as one slider for each feature which they could adjust (Figure 8). In total, 6 recruiters participated in the experiment (although one of them only submitted their slider ratings, and not their model judgements).

By adjusting the sliders for each feature, they could distribute the 'relevance points' and thereby indicate which features they considered most important for the given prediction. After submitting their relevance distribution, the recruiters were redirected to a page that again showed the data of the user, the prediction made by the model, this time accompanied by the model's explanation, and the four questions regarding the understandability of the explanations and the usability of the model (Figure 9). Once the recruiters gave a rating to each explanation type, the recruiters would be shown the second example and repeat the steps. Upon having completed the third example, they were informed they were done, after which their results were

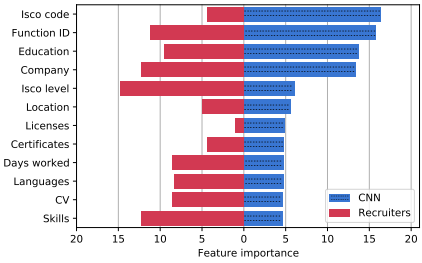
retrieved from the ec2 server and processed using Python.

F Explanation examples

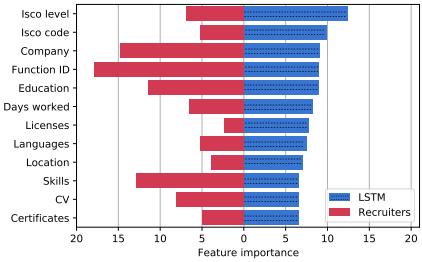
The explanations provided by the three different models for the same candidate can be found in Figures 10, 11, and 12. The correct label for this candidate was *Survey and market research interviewer*.



(a) Distribution of feature importance of the CNN-LSTM.



(b) Distribution of feature importance of the CNN.



(c) Distribution of feature importance of the LSTM.

Fig. 5: Distribution of feature importance of the different models compared to that of Randstad's recruiters. $\mathbf{N = 6}$, averaged over three categories.

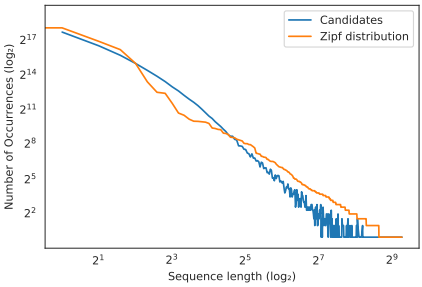


Fig. 6: The full distribution of the job sequence lengths (number of jobs held per candidate). The longest single sequence consisted of 613 jobs. Both axis are in log_2. Distributed according to $Zipf(\alpha = 1.5, n = 613)$.

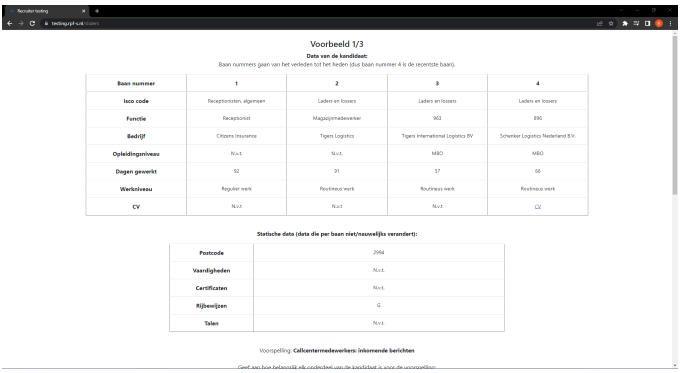


Fig. 7: The interface for observing the users' data. Time series data and static data are shown separately in the two different tables to improve clarity. Below the tables, the label predicted by the model is displayed in bold.

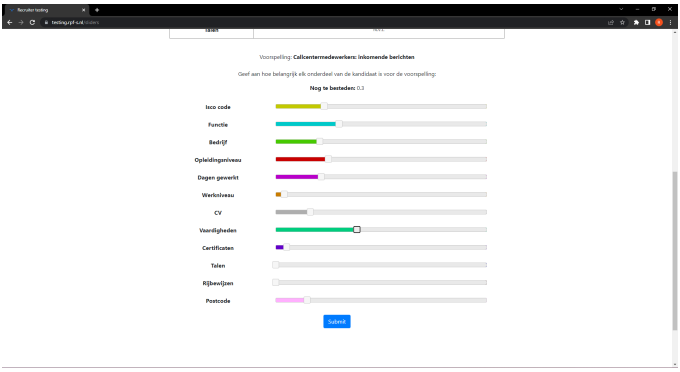


Fig. 8: The sliders used to determine feature importance. At the top, the total amount of 'relevance points' left to spend is displayed in bold. Once this number reaches 0, the sliders can no longer be increased, unless another is decreased.

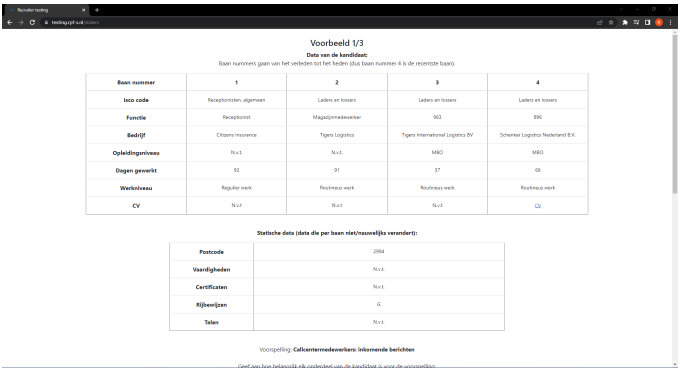


Fig. 9: The interface for judging the models' explanations. By scrolling up, the prediction made by the model, as well as the users' data can be observed (as in Figure 7). A brief explanation on how to interpret the explanations is also provided.

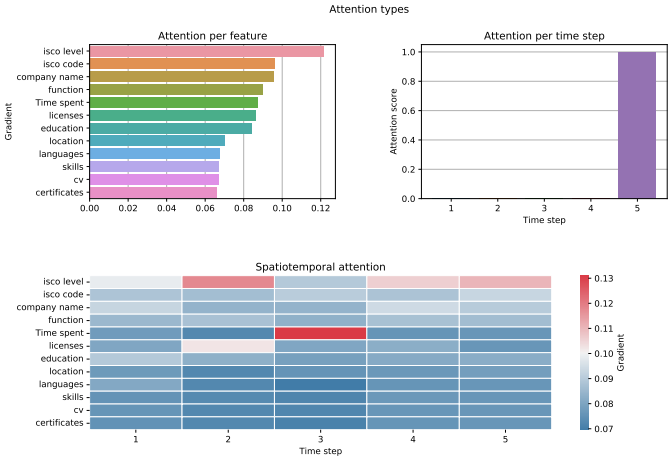


Fig. 10: Explanations provided by the explainable LSTM. Top left: attention per feature. Top right: attention per time step. Bottom: Feature/time step interaction (spatiotemporal attention).

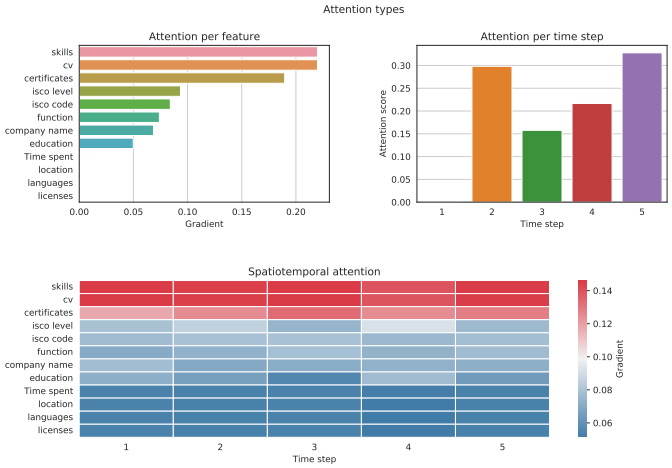


Fig. 11: Explanations provided by the explainable CNN. Top left: gradient weight per feature. Top right: gradient weight per time step. Bottom: Feature/time step interaction (grad-CAM)

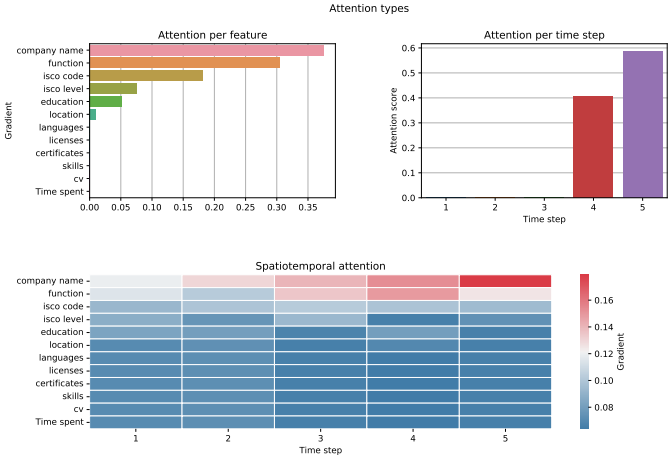


Fig. 12: Explanations provided by the explainable CNN-LSTM. Top left: gradient weight per feature. Top right: attention per time step. Bottom: Feature/time step interaction (guided backpropagation)