Implementing AIRM: A New AI Recruiting Model for the Saudi Arabia Labour Market

monirah Ali aleisa (✉ ma989@sussex.ac.uk)
University of Sussex https://orcid.org/0000-0002-8489-857X

Natalia Beloff
University of Sussex

Martin White
University of Sussex

Research

Keywords: Recruiting, Job Seeker, AI, Natural Language Processing, Clustering Algorithms, BIRCH, RoBERTa, FAISS, Data Lake

Posted Date: October 27th, 2021

DOI: https://doi.org/10.21203/rs.3.rs-965047/v1

Implementing AIRM: A New AI Recruiting Model for the Saudi Arabia Labour Market

Monirah Ali Aleisa[1], Natalia Beloff[1], Martin White[1]

[1]University of Sussex, Falmer, Brighton BN1 9RH, United Kingdom
{Ma989},{n.beloff},{m.white}@sussex.ac.uk

Abstract.

Artificial intelligence is one of the most rapidly evolving technologies today, and it has been used in a variety of application domains. Recently, AI technology has shone in the field of recruiting. Many researchers are working on expanding its capabilities with various applications that will aid in the recruitment process. However, importantly, having an open labour market without a cohesive data centre makes it difficult to monitor, integrate, analyse, and build an evaluation matrix that helps reach the best match of a job candidate to a job vacancy. Moreover, previous research in this domain, where AI serves in the recruiting field, has focused solely on linear and non-linear models that, once trained, lack user preference personalisation. Furthermore, it was more focused on commercial applications than on government services.

This paper implements the AIRM architecture and found impressive results by exploring AI and Natural Language Processing matching job candidates. We used a suitable data repository technique with three processing layers, each with a different appropriate model. First, the Initial Screening layer uses the BIRCH clustering algorithm. The Mapping layer then performs an approximate nearest neighbour search using a Sentence transformer and ranking with a Facebook AI Similarity Search

Finally, the Preferences layer takes the user's preferences as a list and sorts the results using a pre-trained Cross-Encoders model that considers the weight of the more important words. We completed the implementation of AIRM and obtained some promising results. We discovered that when at least one expert agrees with the system's choice, the AIRM achieves an overall matching accuracy of 82%. In the time performance test, AIRM outperforms human performance in terms of task execution time. It completed the task in 2.4 minutes, but humans took more than three days on average, which is essential when pre-selecting a set of candidates and positions. AIRM will empower the Saudi government's immediate and strategic decisions by gaining comprehensive insight into the labour market and speeding up the recruiting process.

Keywords: Recruiting, Job Seeker, AI, Natural Language Processing, Clustering Algorithms, BIRCH, RoB-ERTa, FAISS, Data Lake.

1 Introduction

Hiring and screening curriculum vitae for jobs is a time-consuming and labour-intensive process. Typically, each resume is written uniquely. Before approaching it for the right job, recruiters must read many resumes as texts and comprehend their content. According to Okun's Law, an increase in unemployment in the United States affects the gross domestic product (GDP). Okun's Law states that a 1% increase in unemployment results in a 2% decrease in GDP (Kenton, 2006). According to the General Authority for Statistics, Saudis' unemployment rate in the fourth quarter of 2020 is 12.6 per cent. This high unemployment percentage does not suit a wealthy, prosperous, and developing country with fewer than 25 million people. The Saudi Vision 2030 is trying to address this issue.

Saudi Vision 2030 is a plan announced on April 25, 2016, and coincides with the date set for announcing the completion of the handover of 80 government projects. The plan was organised by the Council of Economic and Development Affairs. Furthermore, it is accomplished in collaboration with the public, private, and non-profit sectors (Affairs_of_V2030, 2016). The Kingdom of Saudi Arabia is putting much effort into this area. However, there are numerous obstacles in the Saudi labour market, each with its own set of challenges. The significance of assisting in the reduction of unemployment is evident here.

Previous research in this domain, where artificial intelligence (AI) serves in the recruiting field, has concentrated solely on linear and non-linear models that, once trained, lack preference personalisation at the user level. Additionally, it served commercial applications more than governments services.

This paper contributes to improving all the above situations by presenting the results of a proposed architecture consisting of a set of AI models and a data repository to build a new AI Recruiting Model for the Saudi Arabia Labour Market (AIRM). AIRM can be used to effectively mimic how the human brain functions when selecting a job as a job seeker or as a recruiter looking for the best candidate to match a vacant position job, taking into consideration user preference. Since all resume data deal with large amounts of natural language as text data, this architecture employs the most recent new profound text semantic understanding technologies, such as contextual embedding, which has been used to capture complex query-document relations.

This paper proposes an algorithm that can retrieve similar candidates for a given job description and vice versa. Since millions of jobs are posted on different platforms leading to a tremendous amount of data, this study also addresses latency issues while retrieving similar job candidates in ample data space.

2 Background Research

This section introduces the research methodology and the best repository techniques that suit the proposed architecture. Finally, AI Models can serve in recruiting field and text similarity.

2.1 Research Methodology

Research methodology is a comprehensive approach that deals with addressing questions like: How are research planned? How are theories created and tested? How are the tests interpreted? (Reich, 1994). Researchers in the field of AI use a variety of methodological approaches. These various methodologies stem from the unique requirements of AI projects and the project's lifecycle, where AI projects are more data-centric than programmatic coding and are implemented in iterative steps (Walch, 2020). Dennis argues that prescriptive methodological analysis is not suitable for AI research. However, researchers should make their methodological perspectives explicit when publishing research results to be better understood.

The research community can learn the skills required to conduct and evaluate research. Despite recent advancements, there are signs that the field of AI still lacks the direction that a clear explanation of such methodological constraints could provide. As can be seen, there is considerable disagreement about what constitutes significant research problems. For instance, Hall and Kibler considered the divergence of opinions gathered in a research survey on knowledge representation. There was no clear consensus in the survey on what knowledge should be represented or what representation entailed. This type of ambiguity can also be seen in selecting appropriate research methodologies, leading to debates such as the ongoing methodological squabble between scruffy and neat views of AI research (Hall & Kibler, 1985).

From the above literature, it is clear that there is no standard methodology for AI research, both academic and industrial, and it depends on the type of problem and the solution for it. Building a robust data-driven prediction model, clustering data, constructing good decision rules, and helpfully visualising high-dimensional data are all things that AI researchers have in common with data mining researchers. These are all iterative tasks that require knowledge and experience to complete. Furthermore, there have been few attempts to develop general methodological guidelines describing the overall process (François, 2008). There are four different data mining guidelines proposals known these days, and each has its charismatics. These methodologies are Fayyad's methodology or knowledge discovery in databases (KDD) (Fayyad, 1996), CIOS methodology, SEMMA methodology, and the cross-industry standard process for data mining methodology (CRISP-DM). A summary comparison of them can be found in Table 1.

Table 1. Difference between methodologies.

Description	KDD	CIOS	SEMMA	CRISP-DM
Objective determination	X	X		X
Data collection	X	X	X	X
Data cleansing	X	X	X	X
Data reduction	X		X	X
Problem reformulation	X			
Data exploration	X		X	
Tools selection	X		X	
Model construction	X	X	X	X
Model validation	X	X	X	X
Result interpretation	X	X	X	X
Deployment	X	X		X

Choosing a research methodology or framework depends on the investigated problem type, as different design tasks have very different characteristics. This study focuses on using AI in the recruiting field, where AI is a relatively new intellectual exploration research field. Therefore, the suitable research type is Design Science Methodology (DSM), with a minimum viable product (MVP) as proof of the new proposed architecture that consists of a set of AI and a data repository to fulfil the research objective. In this research, we use Natural Language Processing (NLP). NLP offers significant mining opportunities in free-form text, particularly for automated annotation and indexing prior to text corpora classification. Limited parsing capabilities can significantly aid in determining what an article refers to. As a result, the spectrum from simple NLP to language understanding can be highly beneficial. NLP can also contribute significantly as an effective interface for stating hints to mining algorithms and visualising and explaining knowledge derived by a KDD system (Fayyad, 1996). When these methodologies are compared to the research objective, it is clear that Fayyad's KDD is the best framework to use. The KDD process is iterative and interactive, with many steps and many decisions made by the user. See Fig. 1.

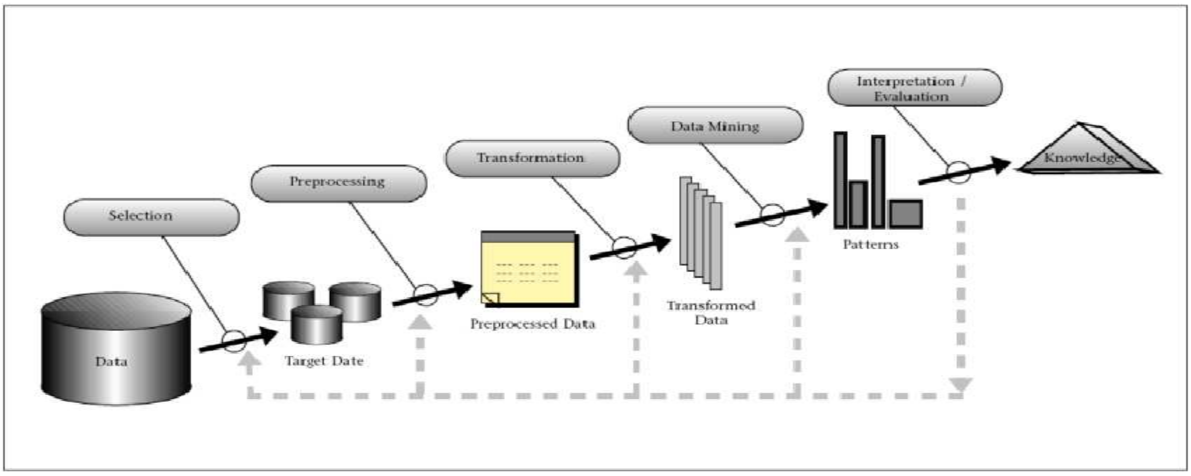

Fig. 1. Steps Involved in a Typical KDD Process (Fayyad, 1996)

The following are the necessary steps of this methodology:

- **Define Business problems:** (Business Objectives) to understand business objectives, then convert them into subproblems to develop the research plan.
- **Data Understanding:** in other words, explore data starts with an initial data collection (Exploratory Statistics, Data Visualization), understanding data after collection, verifying data quality. Further trends and relationships between attributes will be observed via data visualisations.
- **Prepare data:** (Data Cleaning and Pre-processing). This step involves selecting the necessary data, cleaning it (imputation, duplicate, fuzzy matching, etc.), reformatting it to fit, and integrating it once the model is complete. Create the model and choose between model techniques or mixed models.
- **Evaluate the model:** using different evaluation methods to guarantee high accuracy.
- **Deploy:** make others use it to give feedback to enhance the model.

2.1.1. AIRM Research Framework

We proposed the AIRM framework in (Aleisa et al., 2021) based on the knowledge discovery methodology in databases (KDD). It is tailored to the specific problem to which the research is contributing. As shown in Fig 2.

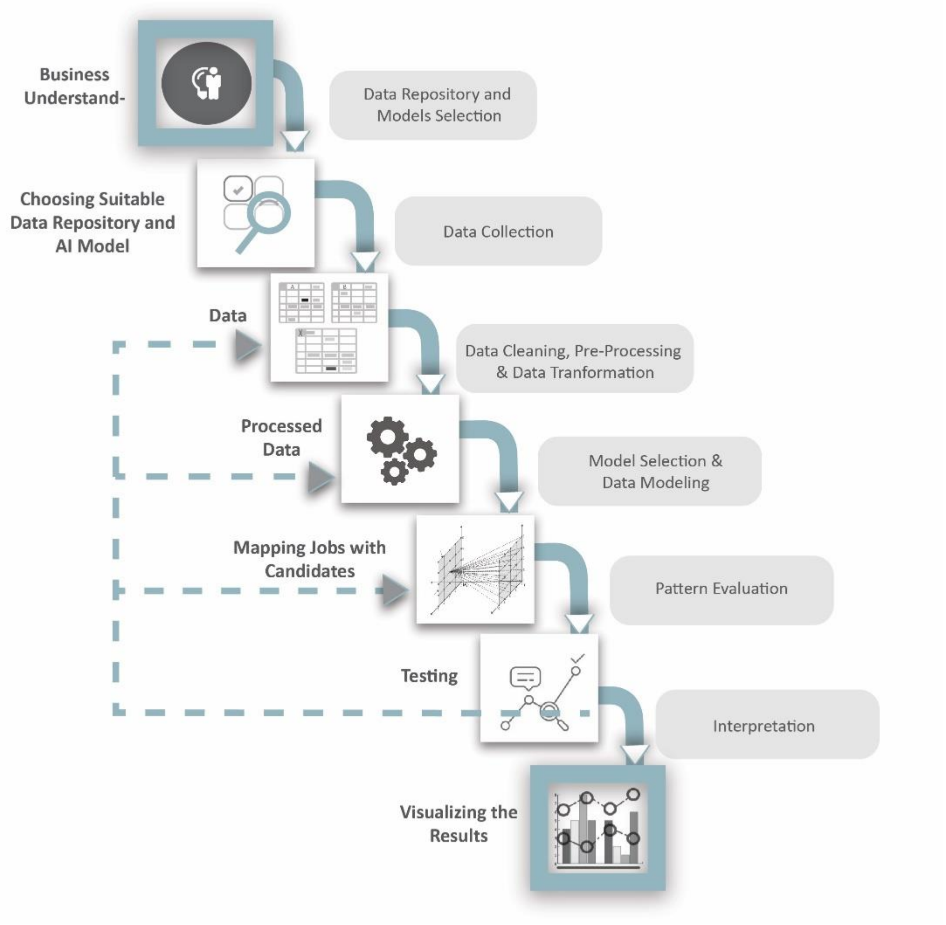

Fig. 2. The AIRM research framework

It starts with business understanding, data gathering, exploring, modelling, evaluation or interpretation, and knowledge discovery. The framework will adopt an iterative approach.

2.1.1.1. Business Understanding

The first step in the adopted KDD process focuses on in-depth analysis of the Saudi labour market, trying to understand the Saudi labour market, national projects that currently support the labour market, labour market growth, the current recruiting system, and the new establishment of the Saudi data and AI authority. This analysis is carried out by reading and analysing all topics related to the development of Saudi labour market projects. Furthermore, a thorough examination of the available data will better understand the Saudi labour market requirements, allowing them to grow and develop productively. The overarching goal of this phase is to answer some intriguing questions:

1. Why hasn't the situation improved significantly despite all of the government's projects?
2. How can AI be effectively used to improve the current Saudi labour market?
3. Which repository is the best for storing national data?
4. What are the most appropriate machine learning algorithms, such as clustering, that can be used to build the proposed architecture that consists of a set of AI models?
5. Finally, how should the new approach be structured?

The proposed AIRM introduce a set of AI models and a state-of-the-art data repository technique that will work with national data in response to the third question. AIRM aim is to connect Saudi job seekers with Saudi labour markets. To address the fifth question, AIRM proposes a framework with a data repository and three layers: an Initial Screening layer, a Mapping layer, and a Preferences layer. The three layers collaborate to determine the best job ID for the job seeker.

2.1.1.2. Choosing Suitable Data Repository and AI Models

This study implements a cutting-edge data repository as well as AI models. It is critical to choose the suitable architecture comprised of appropriate data repositories and AI Models after understanding the business problem and the type of data at hand. For reasons detailed in the proposed AIRM architecture section.

2.1.1.3. The Model Data

The data selected and integrated to implement and test the AIRM architecture are from the labour market field. The data fields for the job information and candidate information are shown in table 2.

Table 2. Job table attributes.

Jobs data	Candidate data
Job title	Candidate personal information
Job responsibility	Candidate qualification
Job qualification required	Candidate achievements
Job skills required	Candidate skills
Job location	Candidate preferred job location
Job industry group	Candidate preferred job industry
Job level	Candidate preferred job level
Job type (e.g., full-time, part-time)	Candidate preferred job type

The data cloning process first collects data from the internet. Data for available jobs and required skills will be cloned from job advertising websites or apps such as LinkedIn, Bayt, and others and cleaned before being fed into the model.

2.1.1.4. Pre-Processing Data

This step involves searching for missing data and removing noisy, redundant, and low-quality data from the data set to improve the data's reliability and effectiveness. Specific algorithms are used for searching and eliminating unwanted data based on attributes related to the application. As part of data pre-processing, relevant features were selected from both job descriptions and candidate profiles. The pre-processing was carried out to ensure that the most relevant attributes were used to find related papers. The job dataset's responsibility, description, and qualification were chosen for further study. Similarly, experience, education, skills and certifications were chosen as relevant features from candidate profiles. In order to ensure that data is free from noise, data cleaning was done as a second step. After exploratory data analysis, undesirable special characters were removed from the text. Numbers and stop words were not removed, considering that transformer models need raw input. Removing stop words can change the semantics of English sentences, and hence they were not excluded during data cleaning. The text was also lower cased as part of pre-processing step.

2.1.1.5. Mapping Jobs with Candidates

This architecture consists of the DL and three layers exploiting machine learning (ML) to extract relevant information from the DL of both recruiters and job seekers to map them. See Fig. 3. The architecture uses a combination of two main technics

- The Rule-based Approach
- Machine Learning Model Approach.

In particular:

1. Prepare the data to be fed to the data chosen algorithms.
2. For coarse clusters, BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies) clustering is used. Ensures that search latency is kept as low as possible. Jobs and Candidates are categorised. There

are, for example, a total of 20 clusters. If a candidate belongs to the fifth cluster, he will be able to find work in that cluster as well. When looking for similar jobs, this eliminates latency (Janrao & Palivela, 2015).

3. Sentence For transfer learning, the Robustly Optimised BERT Pretraining Approach (RoBERTa) model will be used, which has been trained on Natural Language Inference (NLI) and semantic textual similarity (STS) data. This transformer-based model will be used to create embeddings for the job description and the candidate profile. The model is optimised to find similar documents because it is trained on a regression objective in a Siamese network fashion.
4. Once created, the embeddings will be indexed in Facebook AI Similarity Search (FAISS). FAISS is used to perform an approximate nearest neighbour search due to its extremely low latency. FAISS is compatible with both GPUs and CPUs. It can also be easily scaled.
5. The goal is to use FAISS to find ten similar jobs for a candidate and vice versa. FAISS eliminates the computational overhead associated with traditional cosine similarity.
6. Then add the user's preferences using the pre-trained Cross-Encoders model. To rerank the result list, considering the weight of the more important words for users from both sides.

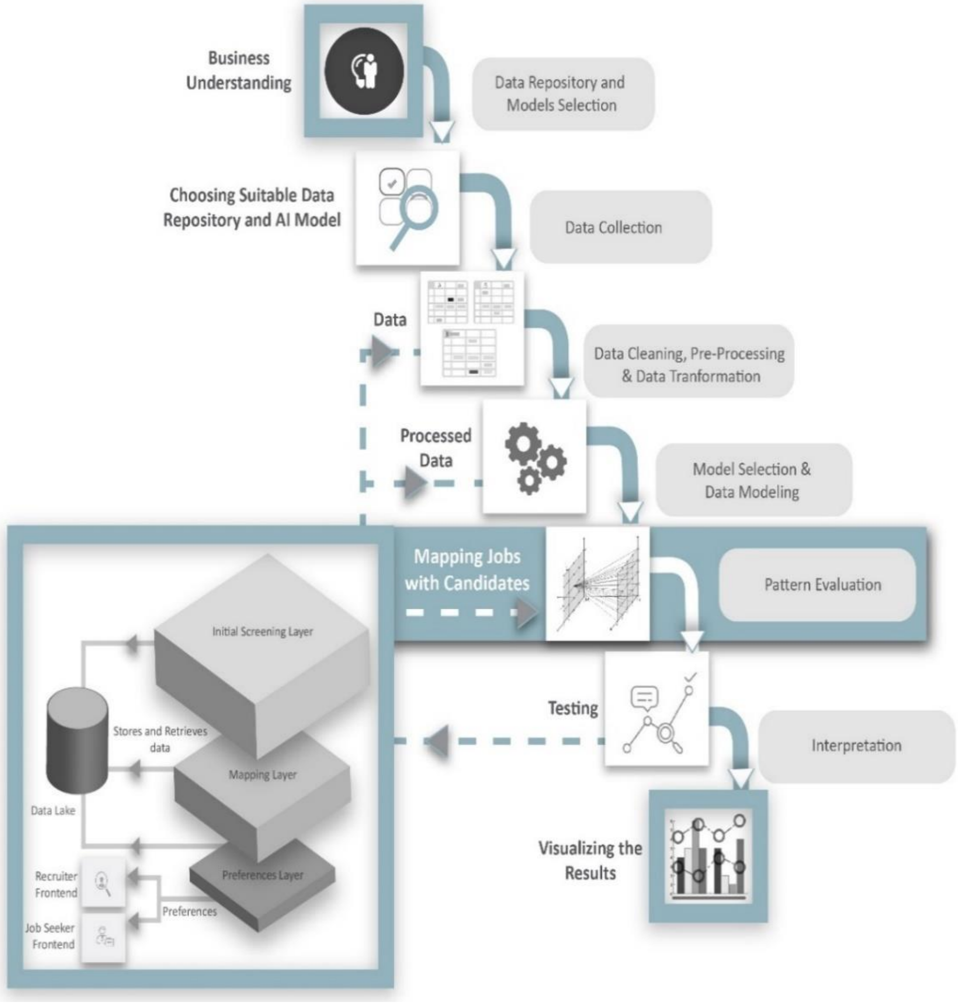

Fig. 3. The three layers in AIRM research framework

Hence, AIRM is a more non-linear method. It deals with text, language processing, and deep learning. It is a tool that helps discover trends from a data set using AI models.

2.1.1.6. Testing

Since AIRM uses a combination of unsupervised ML and AI algorithms. In order to test the performance, a group of external HR domain experts will check output samples and label the result as excellent match, good match, lousy match. Such feedback will help improve AIRM performance.

2.1.1.7. Visualising and discussing the results

Following the acquisition of job mapping, job trend, and patterns via various data model methods and iterations, these patterns must be represented in discrete forms such as bar graphs, pie charts, histograms, and so on, in which the knowledge extracted from the previous step is applied to the specific application or domain in a visual format. This step provides critical assistance and guides the decision-making process.

2.2 Repository Techniques

Getting all the data in one place will support integrating it to allow data engineering to clean it. Then, data scientists can analyse it and apply ML algorithms to the data. This section provides a background of Data Lake (DL) and utilisation.

2.2.1. Data Lake

James Dixon was the first to mention the concept of a DL as a data repository in 2010. He stated that a DL manages raw data as it is ingested from multiple data sources. It does not require cleansed data or structured data (Quix & Hai, 2018). A DL is a daring new approach that harnesses the power of big data technology. It is "A methodology enabled by a massive data repository based on low-cost technologies that improve the capture, refinement, archival, and exploration of raw data within an enterprise" (Fang, 2015). Data are stored in the DL in their original format, whether structured, unstructured, or multi-structured. Once data are placed in the lake, it is available for analysis (Khine & Wang, 2018). A comprehensive discussion about DL, with its architecture build characteristics, data types, and its use, and how suitable it is for AIRM, was presented in our previous work in (Aleisa et al., 2021). In this paper, we implemented the proposed architected and analyse the results.

2.2.2. Kylo

This section will go over Kylo, a new open-source DL, and how powerful it is and how well it fits with AIRM. Kylo is a high-performance DL platform built on Apache Hadoop and Spark. It provides a turnkey, business-friendly DL solution, complete with self-service data ingest, preparation, and discovery. It is a web application layer with features tailored to business users like data analysts, data stewards, data scientists, and IT operations personnel. Kylo incorporates industry best practices for metadata capture, security, and data quality.

Moreover, Kylo offers a flexible data processing framework (based on Apache NiFi) to create batch or streaming pipeline templates and enable self-service features without jeopardising governance requirements. It was created by Think Big, a Teradata company, and is used by a dozen major corporations worldwide (Think Big, 2018). The Apache Software Foundation's Apache NiFi project automates the flow of data between software systems. Using the Extract, Transform, and Load (ETL) paradigm (Apache NiFi Team, 2021). Think Big has reaped significant benefits from the open-source Hadoop ecosystem and has chosen to open-source Kylo to give back to the community and improve value. See Fig.4.

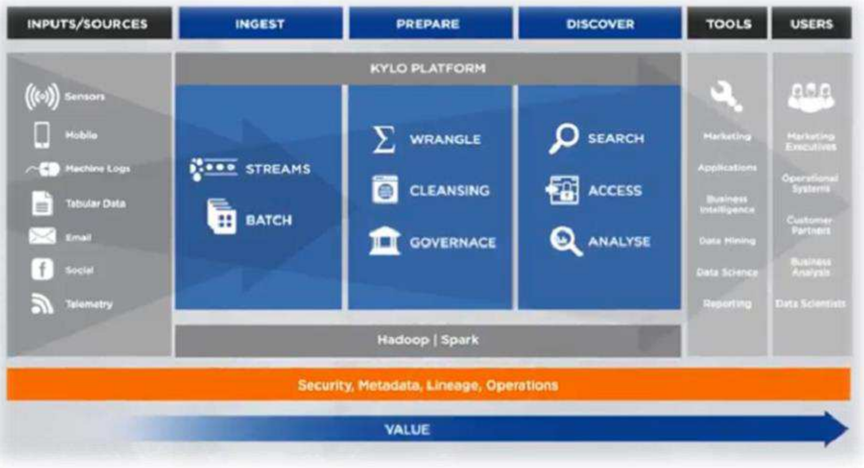

Fig.4 Kylo architecture (Think Big, 2018)

In most cases, the workload is allocated to the cluster with the most processing power. Kylo orchestrates pipelines using Apache NiFi. With 180+ built-in processors, NiFi can connect to various sources and conduct lightweight conversions on edge. Kylo's NiFi processor extensions can call Spark, Sqoop, Hive, and even traditional ETL tools. SQL transformations are the focus of ETL solutions, which use their unique technology. The majority of data warehouse transformations are focused on importing normalised relational schemas like a star or snowflake. ELT tends to follow Hadoop data patterns (Think Big, 2018).

Kylo's added values are:

- Kylo is a modern web application that runs on a Spark & Hadoop cluster's Linux “edge node.” Kylo includes a variety of custom routines for DL activities that use Spark and Apache Hive.
- Kylo's scheduling and orchestration engine are Apache NiFi, which provides an integrated foundation for developing new 200-processor pipelines (data connectors and transforms). Kylo includes a built-in metadata server that is currently compatible with MySQL and Postgres databases. For cluster monitoring, Kylo can connect to Apache Ranger or Sentry, as well as CDH Navigator or Ambari.
- Write-once, use many times, is one of Kylo's additional values. Although NiFi is a robust IT tool for constructing pipelines, most DL feeds only use a few distinct flows or patterns. IT can develop and register a NiFi template as a data processing model for feeds using Kylo.
- Web modules provide important DL functionalities such as metadata search, data discovery, data wrangling, data browsing, and event-based feed execution to connect flows.
- Data feeds can be monitored using the Operations Dashboard UI. It delivers feed health monitoring and related infrastructure services from a central location.

In order to implement AIRM, we do not need expensive or complicated ETL technologies for Hadoop. We use Kylo, as it is more than enough for AIRM. It uses Spark to wrangle and prepare visual data transformations with all added values mentioned above over Apache NiFi.

2.3 AI Algorithms Suitable for the Recruiting Field

This section emphasises a fundamental understanding of ML algorithms required to construct AIRM, such as clustering and NLP algorithms. Before considering any AI algorithms, it is vital to prepare the data set. Some steps are required, such as scaling or normalising the data and imputation of missing values. The feature values of each observation are represented as coordinates in n-dimensional space to calculate the distances between these coordinates. If these coordinates are not normalised, the results may be incorrect. It is also necessary to deal with missing, null, or inf. There are several methods for dealing with such values, such as removing them or inputing them using mean, median, mode, or advanced regression techniques (Rani, Rohil, 2013), We pre-prepared the data for AIRM to ensure optimal results.

2.3.1. Clustering Algorithms

For the reader's convenience, this section will highlight some clustering algorithms to show the purpose of selecting a particular clustering algorithm. Clustering is the process of discovering a series of trends in a set of data. It produced positive results, especially in classifiers and predictors (Homenda & Pedrycz, 2018). Entities in one cluster should be as dissimilar to entities in another cluster as possible. Clustering is an unsupervised approach that can spot erroneous class names, outliers, and mistakes (Frades & Matthiesen, 2010). There are two types of clustering: hard clustering and soft clustering (SHARMA, 2019). There are several methods for measuring the distance between clusters to determine clustering rules, commonly referred to as Linkage Methods (Rani1 &#38; Rohil, 2013). K-means or partition-based algorithms (Aggarwal & Reddy, 2014), hierarchy-based algorithms, fuzzy theory-based algorithms, grid-based algorithms, and density-based algorithms are the five popular clustering algorithms (Aggarwal & Reddy, 2014). K-means is the most popular of all clustering algorithms.

BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies) is a clustering algorithm that excels at clustering extensive datasets. BIRCH handles large datasets by producing a more compact summary that retains as much distribution information as possible before clustering the data summary rather than the original dataset. The cost of BIRCH I/O is proportional to the size of the dataset: a single scan yields good clustering, and one or more additional passes can optionally be used to improve the quality even more. By evaluating its running time, memory usage, clustering quality, stability, and scalability and comparing it to other existing algorithms, we argue that BIRCH is the best available clustering method for handling enormous datasets. Its architecture also supports parallel and concurrent clustering, and performance can be tuned interactively and dynamically based on data knowledge gained (X. Zhang et al., 2016). Shown in Table 3 are some comparison points between the clustering algorithms.

Table 3: Clustering algorithms comparison (Janrao & Palivela, 2015).

	# Of classes before computations	Advantages	Disadvantages
K-means or partition-based	Mandatory	When the number of variables is large, K-Means is usually faster than hierarchical clustering when k is small. It results in tighter clusters.	Predicting K-values is difficult. This algorithm is ineffective for globular clusters. k-means works best for images with spherical clusters with the same variance.
Hierarchical	Optional	Using well-scattered representative points and centroid shrinking can identify clusters with non-spherical shapes and wide size variations. It is capable of handling large databases by combining random sampling and partitioning. Outliers do not pose a problem	Inability to make changes once the decision to split/merge has been made. There is a lack of interpretability in cluster descriptors. It is prohibitively expensive for high-dimensional and massive datasets. The curse of the dimensionality phenomenon severely reduces effectiveness in high-dimensional spaces.
Fuzzy or soft k-means	Mandatory	Unsupervised Converges	Extended computational time Sensitivity to the initial guess (speed, local minima) Sensitivity to noise and One expects low (or even no) membership degree for outliers (noisy points) and post clustering records will have to be smoothed
Density-based	Mandatory	DBSCAN does not require information about the number of clusters in the data. Noise resistance requires only two parameters and is unaffected by order of the points in the database. It can accommodate a wide range of cluster	-DBSCAN is not entirely deterministic. DBSCAN's cluster descriptors are inadequate. When dealing with clusters of varying densities or high dimensional data, DBSCAN does not perform well. These algorithms typically rely on a nonuniform

		shapes and sizes. Outliers do not pose a problem.	distribution of data records (density differences) to find clusters.
Grid-based	Mandatory	It is identifying subspaces of high-dimensional data space that allow for better clustering than the original space automatically. It is possible to think of it as both density-based and grid-based. It is unaffected by order of records in the input and does not assume a canonical data distribution. It scales linearly with input size and has good scalability as the number of data dimensions increase	The accuracy of the clustering result in CLIQUE may be compromised at the expense of the method's simplicity. These algorithms typically rely on a nonuniform distribution of data records (density differences) to find clusters.
BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies)	Optional	Extremely effective. Flexibility in terms of granularity is built in. Ideal for point-to-point linkage problems. It is local in the sense that each clustering decision is made without first scanning all data points or all existing clusters. It employs measurements that reflect the natural closeness of points and can be incrementally maintained throughout the clustering process. In sparse regions, outliers are treated as such and are optionally removed. IT makes full use of available memory to generate the best possible sub-clusters. The clustering and reducing process is organised and distinguished by the use of an in-memory, height-balanced, densely populated tree structure. As a result of these characteristics, its running time is linearly scalable.	Changes cannot be made once the decision to split/merge has been made. Cluster descriptors have a lack of interpretability. The ambiguity of the termination criterion. For high-dimensional and massive datasets, it is prohibitively expensive. The curse of dimensionality phenomenon causes severe effectiveness degradation in high dimensional spaces.

AIRM will focus on BIRCH. The similarity between the clusters is calculated from the dissimilarity measures like the Euclidean distance between two clusters. So, the more significant the distance between two clusters, the better the result is (Pathak, 2018) BIRCH is suitable for AIRM because of four reasons mentioned in this paper in section 4.

2.3.2. Natural Language Processing

In this paper, AIRM will use two State of the Art (SoA) tasks: text similarity for the similar candidate to job matching and vice versa, and fast retrieval of similar documents. Text similarity refers to the comparison of vectors in hyperdimensional space. These vectors are of two, sparse and dense vectors. Sparse vectors deal with Bag of Words (BoW) and TF-IDF (Term Frequency — Inverse Document Frequency) (Qaiser & Ali, 2018). There are few drawbacks to the TF-IDF approach. Since it is a statistical method, it fails to account for the order of words. Another drawback with such an approach is that it does not consider the semantics of a text. These problems of statistical methods can be overcome by other algorithms such as Word2Vec (Mikolov et al., 2013). Word2vec outperforms statistical methods such as BoW and n-gram models on various tasks. It also overcomes the affliction of dimensionality associated with statistical methods like TF-IDF and BoW. Word2vec work with a feed-forward neural network with a language modelling task and optimisation techniques such as Stochastic gradient descent. Word2vec has some drawbacks as it is not able to handle words with different context. It has same representation for words with different context. For instance, if we consider two sentences, *I want to open a bank account* and *I want to sit near a river bank.* Here, word *bank* has different meaning in both sentences and Word2vec model is not able to capture this contextual information. Also, Word2vec does not provide sentence embeddings. For sentence embeddings, word vectors need to be averaged. All these drawbacks can be well handled by transformer based models.

Transformer models were introduced in 2017. The NLP field was revolutionised when Google introduced BERT (Devlin et al., 2019). Transformer models like BERT achieve SoA results on the majority of NLP tasks. The transformer is a model architecture that completely avoids recurrence, favouring drawing global dependencies between input and output. It can be trained significantly faster than architectures based on recurrent or convolutional layers (Vaswani et al., 2017). They even outperform sequence to sequence models like RNN (Recurrent Neural Networks). RNN-based models such as BiLSTM and GRU read input data sequentially. Another problem associated with RNN-based models is that their performance degrades as the input sequence length increases. Transformer based models use an attention mechanism where each data is read once as a sequence of words. BERT is pre-trained on a massive corpus. Pre-training of such models require extensive data as well as substantial computation resources. Pre-training of BERT consist of Masked Language Modelling (MLM) and Next Sentence Prediction (Devlin et al., 2019). Other algorithms can overcome BERT problems like RoBERTa.

The transformer-based RoBERTa model was introduced in 2019 and outperformed BERT in many NLP tasks (Liu et al., 2019). RoBERTa has the same architecture as that of BERT model. However, there are some differences in training the RoBERTa model. RoBERTa is pre-trained with larger batches and on a more extensive corpus. As compared to BERT, the RoBERTa model does not use Next Sentence Prediction (NSP) for the pre-training step (Liu et al., 2019). If NSP is not there, we do not need additional pre-training steps, and the model converges faster with better results. Due to it's better performance over BERT, RoBERTa was chosen over BERT for generating embeddings. In this paper, the RoBERTa base was chosen. The RoBERTa base has 12 encoders stacked over each other compared to RoBERTa large, which has 24 encoders. Fig. 5 shows an architecture of an encoder in RoBERTa model. It consists of self-attention and feed-forward forward layers.

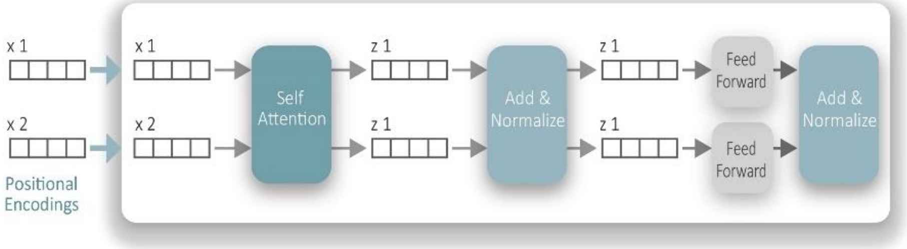

Fig 5. RoBERTa encoder architecture (Vaswani et al., 2017)

For the reranking process, MS MARCO Cross-Encoders is used. MS MARCO is a Machine Reading Comprehension dataset on a vast scale built by Microsoft AI & Research (Bajaj et al., 2018). MS MARCO works as a large-scale information retrieval corpus created with the help of the Bing search engine and real-world user search queries. The cross-encoder model trained on MS MACRO can be used for semantic search; the model will find passages relevant to the search query. The training set contains about 500k samples, while the corpus has over 8.8 million paragraphs (SBERTdocuments, 2021). This reranking technique is used in the third layer of the AIRM architecture.

3 AIRM Architecture

The proposed solution architecture AIRM is the topic of this section. It is constructed to create a proof of concept or a solid minimal viable product (MVP). Our system extracts valuable information from the existing DL using data from both sides, recruiters and job seekers. The Initial Screening layer, the Mapping layer, and the Preferences layer make up the AIRM architecture. The three layers work in sequence to match the job seeker with the best job ID. It retrieves and sends data from/to the DL. See Fig. 6.

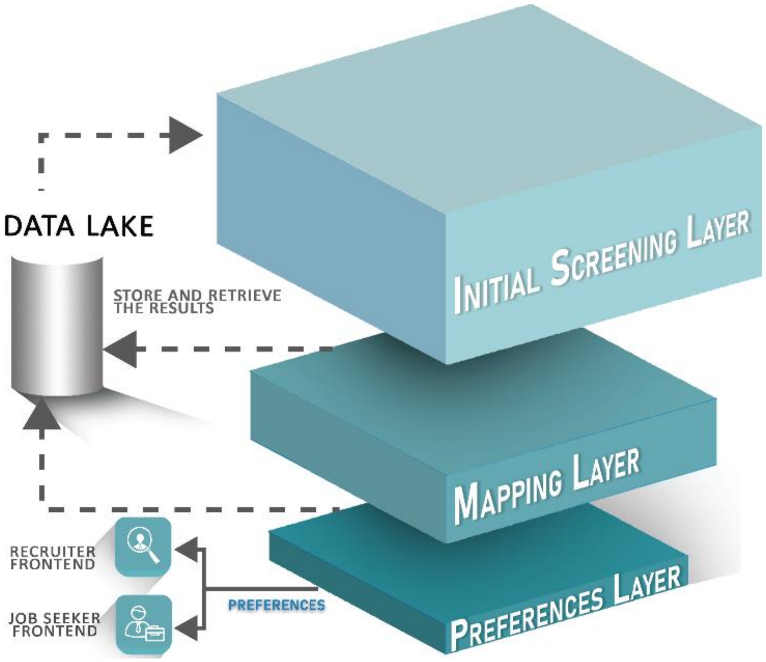

Fig. 6. Proposed AI Recruiting Model (AIRM)

The AIRM model makes the following requirements:

- A national recruiting platform already exists.
- All recruiter's and candidate's data are clean, prepared for analysis, and stored in the DL.
- Candidates can be tagged to one job or more, and the Job ID can be tagged to one candidate or more.
- A comprehensive directory of job specialities in the DL, with instant updating.

3.1 Initial Screening Layer

This layer will work as a preparation phase. It will use the BIRCH cluster algorithm to build cluster groups job specialisations, which will enable the second layer to treat each cluster speciality separately. This layer's input is from both sides, the recruiter's data and job seekers' data. From the recruiter's data, the industry name, job level, job location, the employment period is full-time or part-time. The job seekers' side is considered. Other required data includes industry, location, employment period, and if it is full-time or part-time. The AI model will set these groups and their ID. The result will be stored as a data frame in the DL. It will be an iterative stream process that considers the immediate changes from the user profile. This layer will reduce the AIRM model's computational requirements for the next layer to enable the Mapping layer to work only with the needed group ID.

Hierarchical clustering is a common form of unsupervised learning. It is suitable for this new proposed model, compared with other clustering algorithms. The similarity between the clusters is calculated from the dissimilarity measures like the Euclidean distance between two clusters. So, the larger the distance between two clusters, the better (Pathak, 2018). The data set must first be prepared before the data clustering process can begin. Scaling or normalising the data and missing value imputation are necessary steps in this layer. The feature values of each observation are represented as coordinates in n-dimensional space (n is the number of features), and the distances between these coordinates are calculated to normalise the data. Fig. 7 shows the workflow in this layer..

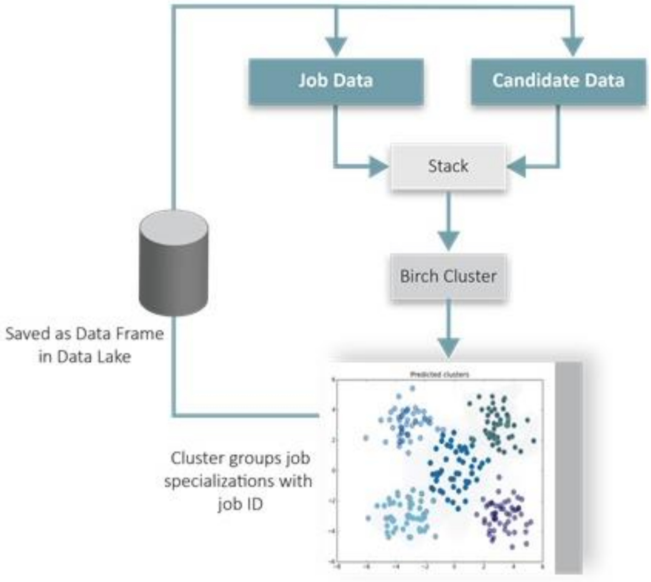
Job Data
Candidate Data
Stack
Birch Cluster
Saved as Data Frame in Data Lake
Predicted clusters
Cluster groups job specializations with job ID

Fig. 7 Tasks in initial screening layer

The first task in this layer is to stack the two data frames over each other. The second task is using TF-IDF vectorisation to convert the text to numeric representation. Here TF-IDF has been chosen as in this case count vectoriser approach will perform better due to nature of the required filters. Then the third task is to feed these vectors to the birch algorithm for the clustering task, then send the clustered groups to the DL. The code we used is available at:

https://github.com/AleisaMonirah/Initial_Screening_Layer/blob/main/Initial_Screening_Layer.ipynb.

3.2 Mapping Layer

This layer will deal with the job groups that have been grouped, and both sides are given a cluster-ID. It uses the Python Natural Language Toolkit (NLTK) library to parse the critical skills and responsibilities from the job description and the qualification, experience, and obligations that the job seeker is willing to take from the job seeker data and vice versa. We have some improvements in this layer after we start implementing it. This layer, mentioned in our last paper (Aleisa et al., 2021), implements a dictionary of keywords with a unique ID by using RNN. Then store in the DL and retrieved when needed. After that, Word2Vec is applied to convert the words in the job description and the job seekers qualification to numeric values, which is vectors and then check the similarity of the words in the job seekers qualification to the dictionary of words built for each job ID. This calculation is for both sides. It is not a redundancy task. The preference from both sides has been added in the next layer. Any outliers are detected and removed. Then the score of each job ID and job seeker ID is sorted in a data frame in the DL. The improvement part is that we used RoBERTa instead of RNN. Due to how robust it is, as mentioned in section 2.4.2 Vector Space Model. A high-level design embeddings generation is shown in Fig. 8.

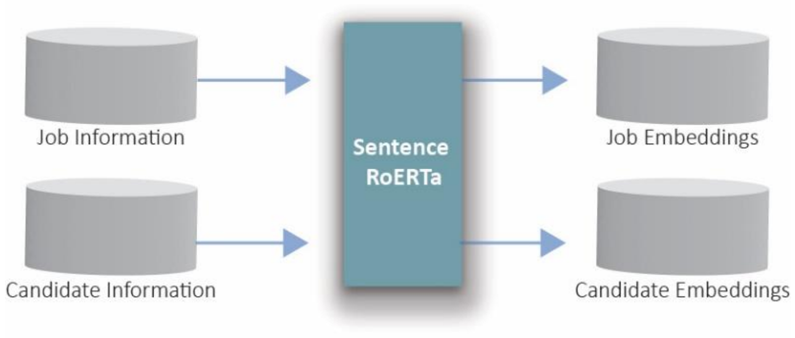

Fig. 8 High-level design – Embeddings Generation

Fig.9 shows how input is fed into RoBERTa model. The input text is tokenised using Roberta specific tokeniser. RoBERTa uses a byte-level BPE tokeniser. The tokenised text is concatenated with unique tokens such as [CLS] (unique classification token) and [SEP] (start of a sentence token) at the beginning and end of sentences. It is done to ensure that model understands the beginning and end of the text. Positional embedding is also appended to input tokens to keep track of the distance between different tokens. The output of the RoBERTa model consists of vectors corresponding to each input token, including [CLS] and [SEP]. Each output vector has a dimension of 768. These vectors can be used for downstream tasks such as text classification, similarity, question-answering, and clustering.

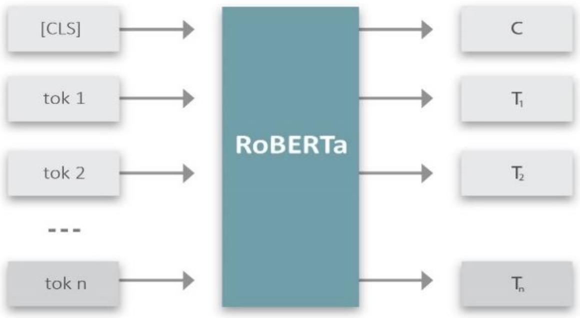

Fig.9 RoBERTa input schema (Narayanaswamy, 2021)

Since the length of input tokens varies in each document, the number of output vectors also varies. In order to overcome this, RoBERTa embeddings for whole input can be averaged, or an output vector corresponding to [CLS] token can also be chosen for further analysis. However, one study shows that sentence embeddings obtained from averaging embeddings or [CLS] token are infeasible to be used with cosine similarity measures due to lousy performance (Reimers & Gurevych, 2019).

This research aims to find similar job-candidate pairs, embeddings needed to be optimised specifically for textual similarity tasks. For this purpose, sentence transformers were used in this research. Sentence transformer consists of transformers based Siamese network. Siamese networks are neural networks with several sub-networks that have the same architecture and weights. In this research, this sub-network refers to RoBERTa model. See Fig. 10.

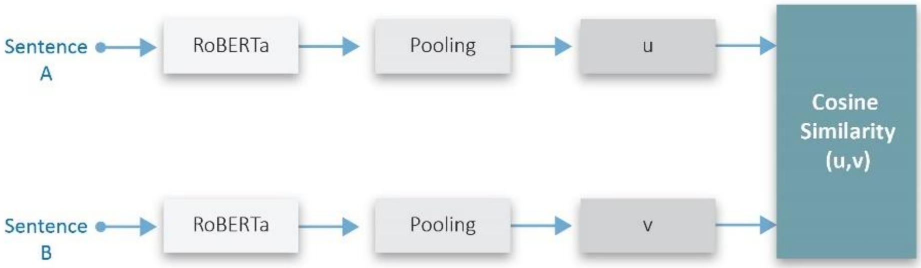

Fig. 10. Siamese Network architecture (Reimers & Gurevych, 2019)

Input sentences are fed to the RoBERTa model, sharing the same weights. The number of output vectors from models is different, depending on the length of input text. Therefore, a pooling strategy is used to obtain sentence embeddings. These embeddings are used to optimise regression function, which is cosine similarity. Study shows that sentence transformers have SoA performance on STS-benchmark dataset (Reimers & Gurevych, 2019). This study's job and candidate data as pre-processed in section 3.3 are fed into this sentence transformer model. The output consists of 768 dimensions embedding corresponding to each job and each candidate.

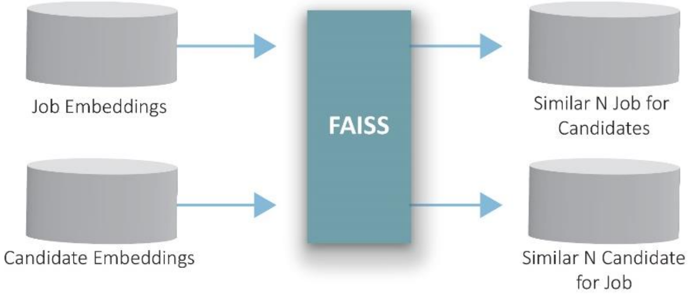

Fig. 11 High-level design – Similar Job-Candidate retrieval

Similar job-candidate retrieval architecture is shown in Fig. 11. The traditional cosine similarity can be very slow as the order of time complexity is O (m*n), where m and n are the number of documents to be compared for cosine similarity. In order to tackle this problem, FAISS (Facebook AI Similarity Search) is used for similarity search. FAISS is a similarity search algorithm developed by Facebook (Johnson et al., 2019). FAISS uses a data structure called index. It can scale up to billions of vectors that possibly do not even fit in RAM. It also provides support for GPU supported document retrieval. It solves the scalability and latency issues as explained above. FIASS uses Euclidean distance to perform each operation on given embeddings. The output from FAISS is top N similar documents (job or candidate). Apart from similar documents, it also provides distance scores that are inversely proportional to similarity: higher the Euclidean distance, lesser the similarity and vice versa.

In the AIRM study, a FAISS index is created as a second task in the mapping layer, and all the embeddings as generated vector space models are added to this index. Once the index is fixed, similar N jobs can be found out for a given candidate along with a similarity measure score. Similarly, top N candidates can be found out for a given job posting. The code we used is available at:

https://github.com/https://github.com/AleisaMonirah/Mapping_layer/blob/main/Mapping_Layer.ipynb

3.3 Preferences Layer

This layer adds the user's preferences using the pre-trained Cross-Encoders model to rerank the result list, while considering the weight of the more important words for users from both sides. Pre-trained Cross-encoders take a text pair as input and provide a score ranging from 0 to 1. They do not compute embeddings for particular texts or work for individual sentences (SBERTdocuments, 2021). Cross encoders are different from bi-encoders in a way that both sentences are passed through the model simultaneously, whereas, in bi-encoder architecture, sentences are passed separately from models sharing the same weights as shown in Fig. 10. Cross encoders are suitable for tasks when similarity needs to be computed between pre-defined datasets. However, cross encoders do not provide embeddings and hence are not suitable for tasks like clustering. Bi-encoders are more suitable for tasks such as information retrieval, semantic search as embeddings can be computed with the help of encoders which are optimised for regression loss.

The user (recruiter or job seeker) will enter keywords as preferences for the degree, the experience, and general preferences. The model will give scores for the reranking, and these scores are combined and normalised to give the final output.

For example, if a recruiter is looking for someone who has worked in an academic field and has published papers or research, the recruiter will prioritise the candidate who has published papers or research. If the recruiter wants this candidate with a specific year of experience, the recruiter will enter the years to give more weight to a candidate with such experiences. Then Cross-Encoders will be applied, and the result will be stored back in the DL. The code we used is available at:

https://github.com/AleisaMonirah/Preference-Layer/blob/main/Preference%20Layer.ipynb

4 Justification of the Implemented Solution

The following components were chosen over others due to given reasons:

1. Kylo is a self-service IT platform that provides robust pipelines as templates and allows users to create feeds from these data processing patterns. Critical Operations activities, including feed monitoring, troubleshooting, and service level measurement, are all provided by Kylo. Kylo's APIs and plug-ins are designed to be expandable, which will appeal to software engineers.
2. The BIRCH clustering is suitable for AIRM because of four reasons:
   - BIRCH clustering does not try to make clusters of the same size as k-mean clustering. The group size will vary in the Saudi jobs data set, and we do not aim to have the same group size. Instead, we need to explore the actual group size in each job group (T. Zhang et al., 1997).
   - BIRCH clustering does not require the number of clusters as an input parameter. In the Saudi jobs data set, the data is significant, and it will change all the time. We cannot decide the number of clusters at the beginning of the algorithm. So hierarchical clustering helps to take away the problem of having to pre-define the number of clusters (Pathak, 2018).
   - BIRCH clustering can virtually handle any distance metric (SHARMA, 2019).
   - BIRCH has been proposed for minimising the running time of clustering. BIRCH incrementally clusters enormous datasets whose sizes are much greater than the amount of available memory. The clustering process is performed by constructing a height-balanced tree (T. Zhang et al., 1997).
3. Sentence transformers capture semantic similarity better than other methods because it is trained in Siamese network fashion. The output loss is cosine similarity which is optimised. In this way, the model learns to capture similarity better than other models and perform SoA on the STS benchmark dataset.
4. In order to compute the cosine similarity between embeddings, traditional cosine similarity leads to enormous computation resources and has a time complexity of O(n*m). This complexity is reduced by using index-based data structures, which FAISS inherently supports. In this way, similar documents can be retrieved very fast. FAISS is also scalable to up to billions of vectors and can search similarity even for embeddings that do not fit in RAM.

5 Results

AIRM was evaluated as an experiment on 100 jobs that were randomly sampled from the job database. Tasks show a superior quality while being more parallelisable and requiring significantly less time. For each job posting, its top 10 similar candidates were retrieved using the algorithm proposed in Section 3.

The recall is a more preferred metric for the task of information retrieval. However, due to the unavailability of any labelled data, it becomes tedious to use these metrics. Also, recall can be increased by returning a higher number of similar candidates/jobs. In order to ensure the quality of similar candidate retrieval, the top 10 similar documents were retrieved. Due to these reasons, accuracy was chosen as a preferred metric for evaluation as compared to recall.

We evaluated AIRM by using two metrics first, the accuracy and then the time. Accuracy for each category was calculated with the help of the below equation:

$$\text{Accuracy} = \frac{\text{Number of correct candidates retrieved by the algorithm}}{\text{Total number of candidates retrieved by the algorithm}}$$

Three human experts in HR and recruiting worked manually to evaluate the results for 100 randomly sampled jobs. Human intervention was required to assess the algorithm's performance for two reasons: firstly, we are using unsupervised algorithms; secondly, choosing a candidate or a job is a subjective judgement.

For the evaluation task, the predictions were labelled as category 3, if all of the three evaluators agreed on them; category 2, if two evaluators agreed on them; and as category 1, if only one evaluator thinks it matches; otherwise, we labelled them as category zero, as shown in table 4.

Table 4. Example of AIRM Results Job-to-Candidate.

Job	Candidate	HR1	HR2	HR3	Final
Job1					
	Similar Candidate1	Good match	Good match	Good match	3
	Similar Candidate2	Good match	Good match	Good match	3
	Similar Candidate3	Good match	Good match	Good match	3
	Similar Candidate4	Good match	Good match	Good match	3
	Similar Candidate5	bad match	Good match	bad match	1
	Similar Candidate6	Good match	Good match	Good match	3
	Similar Candidate7	Good match	bad match	Good match	2
	Similar Candidate8	Good match	Good match	Good match	3
	Similar Candidate9	Good match	Good match	Good match	3
	Similar Candidate10	Good match	Good match	Good match	3
•					
•					
•					
•					
Job10					
	Similar Candidate1	Bad match	Bad match	Good match	1
	Similar Candidate2	Bad match	Good match	Bad match	1
	Similar Candidate3	Good match	Good match	Good match	3
	Similar Candidate4	Good match	Bad match	Bad match	1

	Similar Candidate5	Bad match	Bad match	Bad match	0
	Similar Candidate6	Good match	Good match	Good match	3
	Similar Candidate7	Bad match	Bad match	Bad match	0
	Similar Candidate8	Good match	Good match	Good match	3
	Similar Candidate9	Good match	Good match	Good match	3
	Similar Candidate10	Good match	Good match	Good match	3

The complete result is available at:

https://github.com/AleisaMonirah/AIRM-Results-Job-to-Candidate/blob/main/Mapping_results.pdf.

From the AIRM above results, 61 per cent match falls in category three, 5 per cent match falls in category two, 16 per cent match fall in category one, and 18 per cent falls in category zero. See Table 5.

Table 5. AIRM Results Job-to-Candidate.

	Categories	Per cent
Three agreed	3	61
No one agreed	0	16
One agreed	1	18
Two agreed	2	5
Total		100

As demonstrated in Table 5, AIRM system gives overall 82% accuracy of matching with at least one expert agreed with the system's selection. Therefore, we consider AIRM system to be suitable for the automatic pre-selection of candidates matching the job description, ready for further refinement by the human experts.

In a test that contains 1990 records of jobs and 16171 records of a candidate, AIRM exceeds human performance in terms of time. It finished the work in 2.4 minutes, whereas humans spent more than three days on average, which is beneficial in pre-selecting block of candidates and jobs. We are optimistic about the future of AIRM and intend to use it for a variety of purposes. We intend to expand AIRM to solve challenges, including combining two sides of the text and dealing with massive inputs and outputs. One of our future objectives in this research project is to implement it to work with the Arabic language and compare the results with the English language version.

6 Conclusion

This paper continues the work of our recent paper (Aleisa et al., 2021), where we proposed an AIRM architecture in order to assist the labour market. In the current paper, we implemented the proposed architected and found impressive results. We are making use of cutting-edge technologies, such as Kylo, an open-source DL with all of the data-storage qualities that are versatile and rapid to extract and analyse the data. Kylo lays in the heart of AIRM architecture, the three layers of models that are stacked on top of each other feed data into and out of Kylo.

The first layer is called an initial screening layer. It builds groups of jobs from the same industry to gather and give a group ID by clustering them. It uses BIRCH clustering for coarse clusters. Ensures search latency is reduced. Jobs and Candidates clustered together. For example, 20 clusters in total. If the candidate belongs to the 5th cluster, then a job will also be searched for him in the 5th cluster. This ensures latency while searching for similar jobs.

The second layer is called a mapping layer, where the sentence RoBERTa is used for transfer learning. This transformer-based model is used to generate embeddings for job descriptions as well as candidate profiles. Since the model is trained on regression objectives in Siamese network fashion, it is optimised to find similar documents. Once embeddings are generated, then these embeddings are indexed in FAISS. FAISS is used to perform an approximate nearest neighbour search and has very low latency. It is easily scalable as well. Using FAISS, we can find an 'n' number of similar jobs for a candidate and vice versa. FAISS overcomes the computational expense of traditional cosine similarity.

The third layer is the Preferences layer, which will add the preferences as a weight of the word that is more important for both sides. Then the result will be stored back in the DL.

In order to evaluate the algorithm's performance, it was necessary to use human input. As part of the evaluation task, three human experts in HR and recruiting evaluated the results for 100 randomly selected jobs. We considered appropriate an AIRM selection agreement by at least one human expert to accommodate subjective nature of the selection process when performed entirely by human HR and recruitment experts. We found that the AIRM system gives overall 82% accuracy of matching with at least one expert agreed with the system's selection. In the time performance test that contains 1990 records of jobs and 16171 records of a candidate, AIRM exceeds human performance in terms of time of the task execution. It finished the work in 2.4 minutes, whereas humans spent more than three days on average, which is beneficial in pre-selecting block of candidates and jobs. It is the first time a single model has reached a new SoA in terms of the overall performance.

Therefore, we consider the AIRM system to be suitable for the automatic pre-selection of candidates matching the job description to allow human experts to concentrate on more detailed and nuanced consideration of already pre-selected subset of job seekers, thus improving the efficiency of human involvement.

Further work for this research project is to find different types of data sets in order to generalise AIRM for other government uses. Moreover, we plan to implement AIRM using the Arabic language and evaluate its performance.

7 References

Affairs_of_V2030. (2016). *V2030 5th Anniversary - vision 2030*. EconomicAffairs. https://www.vision2030.gov.sa/ar/mediacenter/news/v2030-5th-anniversary/

Aggarwal, C. C., & Reddy, C. K. (2014). *Data clustering : algorithms and applications* (pp. 234–297). Boca Raton, Fla. : CRC Press ProQuest Ebook.

Aleisa, M. A., Beloff, N., & White, M. (2021). AIRM: A New AI Recruiting Model for the Saudi Arabia Labor Market. *Intelligent Systems Conference (IntelliSys) 2021*, *296*, 105–124. https://doi.org/10.1007/978-3-030-82199-9_8

Apache NiFi Team. (2021, July 10). *Apache NiFi Documentation*. https://nifi.apache.org/docs.html

Bajaj, P., Campos, D., Craswell, N., Deng, L., Gao, J., Liu, X., Majumder, R., Mcnamara, A., Mitra, B., Nguyen, T., Rosenberg, M., Song, X., Stoica, A., Tiwary, S., & Wang, T. (2018). *MS MARCO: A Human Generated MAchine Reading COmprehension Dataset*.

Devlin, J., Chang, M.-W., Lee, K., Google, K. T., & Language, A. I. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv*.

Fang, H. (2015). Managing data lakes in big data era: What's a data lake and why has it became popular in data management ecosystem. *2015 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2015*, 820–824. https://doi.org/10.1109/CYBER.2015.7288049

Fayyad, U. (1996). From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, *17*(3). https://doi.org/10.1007/978-3-319-18032-8_50

Frades, I., & Matthiesen, R. (2010). *Overview on Techniques in Cluster Analysis Bioinformatics Methods in Clinical Research*. *593*, 81–107. https://doi.org/10.1007/978-1-60327-194-3

François, D. (2008). Methodology and standards for data analysis with machine learning tools. *ESANN 2008 Proceedings, 16th European Symposium on Artificial Neural Networks - Advances in Computational Intelligence and Learning*, *January 2008*, 239–246.

Hall, R. P., & Kibler, D. F. (1985). Differing Methodological Perspectives in Artificial Intelligence Research. *AI Magazine*, *6*(3), 166–178.

Homenda, W., & Pedrycz, W. (2018). CLUSTERING. In *Pattern Recognition* (pp. 247–273). John Wiley & Sons, Inc. https://doi.org/10.1002/9781119302872.ch8

Janrao, P., & Palivela, H. (2015). Management zone delineation in Precision agriculture using data mining: A review. *ICIIECS 2015 - 2015 IEEE International Conference on Innovations in Information, Embedded and Communication Systems*, *September*. https://doi.org/10.1109/ICIIECS.2015.7193256

Johnson, J., Douze, M., & Jegou, H. (2019). Billion-Scale Similarity Search with GPUs. *IEEE Transactions on Big Data*, *7*(3), 535–547. https://doi.org/10.1109/TBDATA.2019.2921572

Kenton, W. (2006). *Okun's Law Definition*. Investopedia. https://www.investopedia.com/terms/o/okunslaw.asp

Khine, P. P., & Wang, Z. S. (2018). Data lake: a new ideology in big data era. *ITM Web of Conferences*, *17*(December), 03025. https://doi.org/10.1051/itmconf/20181703025

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V., & Allen, P. G. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*, *1907.11692*. https://github.com/pytorch/fairseq

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. *ArXiv*, *1301. 3781*.

Narayanaswamy, G. R. (2021). *Exploiting BERT and RoBERTa to Improve Performance for Aspect Based Sentiment Analysis Gagan Reddy Narayanaswamy*. https://doi.org/10.21427/3w9n-we77

Pathak, M. (2018, July 24). *Hierarchical Clustering in R - DataCamp*. Datacamp. https://www.datacamp.com/community/tutorials/hierarchical-clustering-R#what

Qaiser, S., & Ali, R. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications*, *181*(1), 25–29. https://doi.org/10.5120/ijca2018917395

Quix, C., & Hai, R. (2018). Data Lake. *Encyclopedia of Big Data Technologies*, 1–8. https://doi.org/10.1007/978-3-319-63962-8_7-1

Rani^1, Y., & Rohil, H. (2013). A Study of Hierarchical Clustering Algorithm. In *International Journal of Information and Computation Technology* (Vol. 3, Issue 10). http://www.irphouse.com/ijict.htm

Reich, Y. (1994). Layered models of research methodologies. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing*, *8*(4), 263–274. https://doi.org/10.1017/S0890060400000949

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *ArXiv*.

SBERTdocuments. (2021). *Pretrained Cross-Encoders — Sentence-Transformers documentation*. SBERT.NET. https://www.sbert.net/docs/pretrained_cross-encoders.html

SHARMA, P. (2019, May 27). *Hierarchical Clustering | Hierarchical Clustering Python*. Analyticsvidhya. https://www.analyticsvidhya.com/blog/2019/05/beginners-guide-hierarchical-clustering/

Think Big. (2018). *Kylo Documentation*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 5999–6009.

Walch, K. (2020). *Why Agile Methodologies Miss The Mark For AI & ML Projects*. Forbes Media LLC. https://www.forbes.com/sites/cognitiveworld/2020/01/19/why-agile-methodologies-miss-the-mark-for-ai--ml-projects/?sh=307b979e21ea

Zhang, T., Ramakrishnan, R., & Livny, M. (1997). BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, *1*(2), 141–182. https://doi.org/10.1023/A:1009783824328

Zhang, X., Zhou, Y., Ma, Y., Chen, B. C., Zhang, L., & Agarwal, D. (2016). GLMix: Generalized linear mixed models for large-scale response prediction. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-Augu*, 363–372. https://doi.org/10.1145/2939672.2939684