Enhancing Job-Competency Matching: A Comparative Study of BERT-Based Approaches for Targeted Assessment

Abstract—**Matching job applicants to the appropriate positions is a complex task, where competency assessment plays a crucial role in identifying the most suitable candidates. This study investigates three methods—KeyAlign, PhraseMap, and Direct SBERT—to match exam questions to job descriptions using keyword extraction and semantic similarity techniques. The methods were evaluated on 30 job descriptions categorized into five ITPE topics. Experimental results indicate that Direct SBERT achieves the highest precision, while keyword-based methods provide efficiency in some cases. These findings suggest that hybrid approaches integrating keyword-based and deep learning models may enhance performance. Future work includes expanding the dataset and incorporating additional evaluation metrics.**

Keywords—Job Matching, Competency Assessment, SBERT, Keyword Extraction, Semantic Similarity, Natural Language Processing (NLP)

I. INTRODUCTION

The rapid evolution of the labor market has intensified the complexity of matching job applicants with suitable positions. Traditional hiring methods, such as resumes and interviews, often fail to fully capture an applicant's competencies. This limitation is particularly evident as job roles become increasingly specialized, requiring a more precise evaluation of skills and knowledge.

Furthermore, existing assessment tools are not tailored to the unique demands of each role, leading to potential mismatches between candidates and employers. This inefficiency not only hampers employment outcomes but also undermines trust in the recruitment process.

To address these challenges, this research introduces a dynamic and targeted competency evaluation framework. The proposed system benefits both organizations and job seekers by ensuring a better alignment of skills with industry requirements. Specifically, young graduates often face difficulties transitioning from academia to the workforce due to gaps between their education and market expectations. Our system bridges this gap by enabling candidates to assess their competencies through a structured set of examination questions related to their target roles.

By identifying strengths and areas for improvement, candidates can refine their skills in a focused manner, thereby increasing their employability and preparedness for industry demands.

II. RELATED KNOWLEDGE / ALGORITHM

In the past, many studies have focused on correlating two text sets, such as matching job descriptions with resumes or analyzing data to uncover textual relationships, exemplified by "Job2Vec: Job Title Benchmarking with Collective Multi-View Representation Learning." Additionally, research employs text analysis methodologies like Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) to derive insights from textual data **[1]**.

However, studies in this area often encounter challenges, including insufficient accuracy in semantic matching, difficulties in managing communications within complex settings, and prolonged processing times when analyzing large datasets. To overcome these limitations, recent research has adopted advanced techniques such as SBERT, KeyBERT, and Cosine Similarity, which offer improved performance in analyzing text relationships with greater semantic accuracy.

A. SBERT

SBERT, or Sentence-BERT It is a natural language processing (NLP) model developed from BERT (Bidirectional Encoder Representations from Transformers) to help solve the problem of calculating semantic similarity between texts efficiently and quickly.

SBERT works by transforming text into numerical representations (vectors) that encapsulate its meaning, thereafter comparing these representations to assess the semantic similarity between the texts. In this way, SBERT is suitable for tasks that require matching or comparing the meaning of text, such as finding related information or grouping texts with similar content **[2]**.

We use SBERT to transform exam questions and job descriptions into vectors. Then compare the meaning of the two statements to assess whether the exam is appropriate for the job position.

B. KeyBERT

KeyBERT is a tool that efficiently and precisely extracts keywords from text by leveraging the BERT model to comprehend the contextual meaning of words inside the text **[3]**.

We use KeyBERT to extract significant keywords from job descriptions and examinations to facilitate the matching of topics with similar context or meaning.

C. EmbedRank

EmbedRank is a method for keyphrase extraction that uses sentence embedding models to represent both the document and keyphrase candidates as vectors. The candidates are ranked based on their semantic similarity to the document's representative vector.

We use EmbedRank to extract key phrases from job descriptions. These extracted key phrases are subsequently analyzed to identify their corresponding domains, ensuring that each job description is accurately categorized based on its content **[4]**.

D. Silhouette analysis

Silhouette analysis is a technique for assessing clustering quality by measuring how similar a data point is to its assigned cluster compared to other clusters. The silhouette score ranges from -1 to 1, where higher values indicate well-separated clusters with cohesive grouping **[5]**.

We apply Silhouette analysis to determine the optimal number of clusters in the clustering process.

E. Cosine Similarity

Cosine Similarity is a mathematical technique for quantifying the similarity between two vectors. It is calculated from the angle between those vectors in the same space. If the angle between the vectors is small (or close to 0 degrees) indicates that the two vectors are very similar. which Cosine similarity has the following formula **[6]**:

$$Cosine\ Similarity = \frac{A \times B}{||A|| \times ||B||}$$

When

$A \times B$ is dot product of vectors A and B

$||A||$ is magnitude of vector A

$||B||$ is magnitude of vector B

We use Cosine Similarity to assess the congruence between the exam vector and the Job Description transformed by SBERT, consequently evaluating the alignment of the exam with the job requirements.

F. NLTK

NLTK (Natural Language Toolkit) is Python's library design for NLP(Natural Language Processing). It offers user-friendly interfaces to more than 50 linguistic corpora and resources, including WordNet. NLTK also provides tools for text tasks such as classification, tokenization, stemming, tagging, parsing, and semantic analysis, along with integration with industrial-grade NLP tools and an active user community **[7].**

We use NLTK to remove Stop Words from job descriptions. This step eliminates insignificant words in the text, ensuring that only meaningful terms remain for subsequent processing and analysis.

G. PyPDF2

PyPDF2 is a free and open-source python library designed for versatile PDF manipulation. It supports operations such as splitting, merging, cropping, and transforming PDF pages **[8]**.

PyPDF2 was employed to extract text from PDF exam files, which was then utilized in the matching process.

III. METHODOLOGY

This section outlines the methodology used to develop and evaluate a system for matching job descriptions with exam questions based on their respective domains. The primary objective of this system is to ensure that individuals possess the necessary competencies and knowledge required for jobs that align with their qualification and needs. This system consists of two main parts: Question Repository and Matching System.

The main dataset used in this study consists of approximately 4,010 ITPE exam questions spanning from 2009 to 2024. These questions cover diverse IT domains, including software development, network administration, and database management. The ITPE exam was sourced from the ITPE Question Bank, an official repository provided by the Information Technology Professional Examination (ITPE). This repository contains a comprehensive collection of exam questions designed to assess competencies in various IT-related domains, including software development, network administration and database management. The dataset is stored as PDF files, each containing structured questions along with metadata such as exam titles and question numbers.

For the efficiency of the system, constructing the Question Repository is essential for its development and evaluation. A Firebase real-time database was utilized to store all data in the form of metadata used in the system processing. The following steps summarize the data preparation and storage pipeline, ensuring the dataset is well-structured and ready for processing.

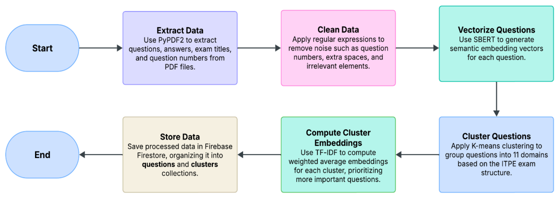

Figure 1: Question preparation processes

The flowchart in Figure 1 represents the question preparation processes, from extracting data from PDF files to storing the processed data in Firebase Cloud Firestore. The key steps in this process are as follows:

1) *Clustering questions:* K-means clustering was applied to categorize the extracted data into groups. The questions were organized into 11 clusters, corresponding to the 11 domains defined in the ITPE exam.
2) *Compute Cluster Embeddings:* In this step, TF-IDF was utilized to compute TF-IDF vectors of the questions within the cluster, which were used to calculate a weighted average of their embedding vectors. This method ensures that the cluster vector reflects the semantic importance of individual questions, with higher priority given to questions with higher TF-IDF scores.
    1. *Store data:* the entire dataset was structured into two collections and stored in Firebase Cloud Firestore as metadata (see Figure 2). The questions collection contains detailed information for each question, while the cluster collection stores summarized data for each cluster.

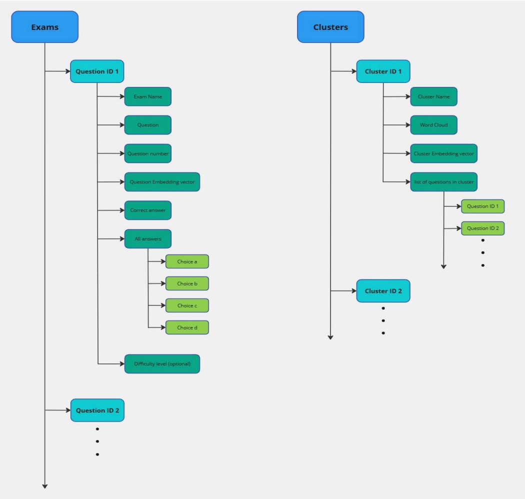

Figure 2: Firebase Structure

This methodology section details a comparative analysis of three distinct methods for matching job descriptions with relevant assessment questions: Semantic Matching with PhraseMap, KeyAlign, and Direct SBERT. Each method leverages unique techniques to analyze job descriptions and identify corresponding assessment content. The purpose of this comparative analysis is to evaluate the effectiveness, accuracy, and computational efficiency of each approach in aligning job requirements with appropriate assessment items.

The input data for this analysis includes job descriptions in text format, detailing specific skills, responsibilities, and requirements for various roles, alongside the number of assessment questions requested by the user.

Example Job description

- Solid experience in PL/SQL development, including code support for functions, procedures, packages, triggers, types, pipelines, and object functions.
- Strong skills in algorithm analysis and performance tuning.
- Experience contributing to database structure design.
- Ability to diagnose and resolve issues in the existing code base.
- Proficiency in accurately implementing and documenting solutions.
- Exceptional analytical and organizational skills.
- A strong problem-solving attitude with the ability to understand end-user requirements.
- Excellent verbal and written communication skills.

Source: Database Developer job description from BillingPlatform, retrieved from [Workable]

This section outlines the processes used by each approach, including input preparation and processing steps of each method.

A. PhraseMap

This method focuses on identifying relevant exams content that corresponds to job description domains by extracting key phrases from the document. EmbedRank utilizes SBERT to create representative vectors for both job descriptions and key phrases, ranking them based on their semantic similarity to identify the corresponding domain. The following steps outline the process of this method.

1. **Extract Key Phrases and Data Preparation**
   - 1.1. **Data Cleaning:** To remove irrelevant characters, Python's regular expressions library was utilized to clean job descriptions by removing numbers, extra spaces, and other unnecessary elements.
   - 1.2. **Generate candidates:** To generate candidates from the job description. The text is split into words, and n-grams (phrases of varying lengths, such as unigrams or bigrams) are generated based on the specified n-gram range. These candidate phrases are later used for matching or further analysis.
   - 1.3. **Stop words removal:** NLTK library was utilized to eliminate key phrases containing stop words. This step ensures that the processed text focuses only on meaningful and relevant terms, improving the quality of subsequent analysis.
   - 1.4. **Filter out duplicate words:** After generating key phrases, duplicate entries were removed to ensure that each key phrase was unique. This step eliminates redundancy, improving the clarity and efficiency of subsequent analysis.
   - 1.5. **Vector and Similarity Calculation:** measure the semantic similarity between key phrases and job descriptions, SBERT was employed to generate representative vectors for both key phrases and the job descriptions. The semantic similarity was then calculated using cosine similarity, enabling an accurate comparison of their meanings.
   - 1.6. **Key phrases selection:** All key phrases were ranked based on their semantic similarity to job description. The top 30% of key phrases were selected for further analysis, with the maximum limit of 15 key phrases to ensure relevance and efficiency in the process.
2. **Cluster Alignment and Question Distribution:**
   - 2.1. **Exam Data Retrieval:** Exam cluster data was retrieved from Firebase Firestore, including the representative vector of each cluster. These vectors were then used to calculate semantic similarity with the selected key phrases.
   - 2.2. **Similarity Calculation:** To determine the similarity between each key phrase and the clusters, the Numpy library was used to reshape the representative vectors of both key phrases and clusters into a suitable format. The cosine similarity metric was then applied to compute similarity scores for each pair. For each key phrase, the cluster with the highest similarity score was identified and selected. The results were stored in a dictionary, which tracks the cluster ID, the count of key phrases associated

with it, its representative vector, and the total accumulated similarity score.

2.3. **Question Distribution:** To allocate the number of questions based on the weighted importance of job description domains, a weighted average was used to calculate cluster weights based on the ratio of key phrases in each cluster to the total key phrases. Questions were distributed proportionally to these weights, with at least one question per cluster. Adjustments were made by adding or subtracting questions incrementally with each cluster. This process incurred that number of questions matched the number of questions required by the user while maintaining proportional distribution.

3. **Exams retrieval:** The entire set of question IDs for each cluster stored in Firebase Firestore was retrieved. The Random library was then used to sample a subset of questions based on the number of questions allocated to each cluster. The details of the sampled questions were subsequently fetched from Firestore for further processing.
4. **Adjustment for Accuracy:** The selected questions are saved in a CSV file, structured for seamless integration into assessment platforms. This file preserves the question text, allowing for easy import and use in evaluation processes.

B. *KeyAlign*

This method aimed to identify exam questions that corresponded to specific job description domains by extracting key phrases. KeyBERT utilized a pre-trained transformer model to extract the most salient keywords from job descriptions. These keywords were then processed using SBERT to generate semantic embeddings for both the extracted keywords and exam questions, allowing for accurate similarity comparison. The following steps outline the process of this method.

1. **Data Preprocessing:** Job descriptions were cleaned using Python's regular expressions library to remove numbers, extra spaces, and irrelevant characters, ensuring that the input was clean and well-structured for keyword extraction.
2. **Keyword Extraction:** KeyBERT, a keyword extraction model, is used to identify the most relevant keywords from the job description. The number of extracted keywords was set to 30% of the total words in the job description, with a maximum limit of 15 keywords.
3. **Cluster Mapping and Question Selection**

3.1. **Retrieving Exam Clusters:** Predefined exam clusters, along with their representative vectors, were fetched from Firebase Firestore. These vectors were essential for measuring the similarity between extracted keywords and existing question clusters.

3.2. **Semantic Similarity Computation:** Each extracted keyword's embedding was compared with the cluster representative embeddings using cosine similarity. The keyword was assigned to the cluster with the highest similarity score. The matching information, including the cluster ID, number of associated keywords, representative vectors, and accumulated similarity scores, was recorded.

3.3. **Cluster Weighting and Allocation:** The relative importance of each cluster was determined based on the proportion of keywords associated with it. A weighting system was used to distribute the selection of questions in a way that reflected the prominence of each cluster.

3.4. **Proportional Question Assignment:** The number of questions allocated to each cluster was determined based on its weight. To ensure alignment with the required total, adjustments were made by incrementally adding or removing questions while maintaining a fair distribution across clusters.

4. **Fetching Exam Questions:** Question IDs corresponding to each cluster were retrieved from Firebase Firestore. A subset of these questions was then selected using a randomized sampling approach, ensuring a diverse and representative selection.
5. **Assessment Generation:** The selected questions are written to a CSV file, formatted for use in assessments. This file contains the question text and can be easily imported into other systems for evaluation purposes.

C. *Direct SBERT*

This method focuses on identifying relevant exams content that corresponds to job description domains by directly comparing job description content with exam content to identify the most relevant matches. SBERT is utilized to generate representative vectors for each section of the job description, while K-means Clustering is used to group them into meaningful domains. The core of this method lies in identifying which exam content clusters are most semantically similar to each domain of job description. The following steps outline the process of this method.

1. **Data Preprocessing:** To prepare for clustering, the job description was divided into individual sections. Irrelevant characters, numbers, and extra spaces were then removed. Subsequently, SBERT was employed to generate representative vectors for each section.
2. **Silhouette analysis:** To determine the optimal number of clusters, silhouette analysis was performed by evaluating the clustering efficiency for cluster counts ranging from 2 clusters to 15. The number of clusters with the highest silhouette score was selected.
3. **Clustering:** K-Means clustering was applied to group individual job description sections into clusters representing the same domain.
4. **Data Retrieval:** The representative vector of each exam cluster was retrieved from Firebase Firestore for subsequent processes, including semantic similarity calculation.
5. **semantic similarity calculation:** To identify only the optimal cluster for each job description cluster, representative vectors of both clusters were reshaped using the Numpy library. Cosine similarity was then applied to calculate semantic similarity between each pair. The exam clusters with the highest similarity score for each job description cluster was selected.

6. **Question Distribution:**
   6.1. **Weight Calculation And Question Allocation:** To determine the number of questions allocated to each job description cluster, The proportion of sections in each job description cluster was calculated by dividing the number of sections in the cluster by the total number of sections. This ratio was then used as a weight to distribute the questions accordingly.
   6.2. **Adjustment for Accuracy:** To ensure the total number of allocated questions matches the user's request, adjustments were made by incrementally adding or subtracting questions from clusters with the highest weights until the desired total was achieved.
7. **Questions Ranking:** All questions from each selected cluster were retrieved and encoded by SBERT to generate a representative vector. The dot product between each question vector and its corresponding cluster vector was computed using NumPy to measure relevance. The questions were then ranked based on these scores, and the top-ranked questions were selected according to the allocated number of each cluster.
8. **Assessment Generation:** The finalized set of selected questions is exported to a CSV file, ensuring a structured format suitable for assessment use. This file includes the question text, making it easy to integrate into various evaluation systems for further processing.

IV. EXPERIMENT

The study compares these methods using 30 job descriptions across five IT domains: Database, Network, Cybersecurity, Software, and IT Business. The evaluation metric used is the Index of Item-Objective Congruence (IOC), rated by two IT experts.

A. Experimental Setup

This study aimed to compare three different methods—Direct SBERT, PhraseMap, and KeyAlign—for matching job descriptions with relevant exam questions. The experiment was designed to assess the effectiveness of these methods in aligning job descriptions with ITPE exam content.

B. Dataset

A dataset comprising 30 job descriptions was used, categorized into five topics based on ITPE exam contents: Database, Network, Cybersecurity, Software and IT Business

Each job description was carefully selected to ensure diversity in job roles and responsibilities within the respective categories. The descriptions were preprocessed by removing unnecessary stopwords and normalizing text to improve keyword extraction and semantic matching performance.

C. Methods

Three different methods were evaluated in this experiment:

1. **Direct SBERT:** The method aimed to directly measure the similarity between job descriptions and clusters using semantic embeddings.
2. **PhraseMap:** The method aimed to extract keyphrases from job descriptions to identify relevant domains and match them with corresponding question clusters.
3. **KeyAlign:** The method aimed to extract keywords from job descriptions to determine relevant domains and align them with corresponding question clusters.

Each method was applied to extract keywords or compute semantic similarity between job descriptions and exam questions. The top-matching exam questions for each job description were then reviewed by experts.

\` Two IT experts manually reviewed the matching results and rated them using the IOC score, which evaluates how well the extracted questions align with job descriptions.

- Rating Scale:
  +1 = Relevant
  0 = Neutral
  -1 = Not Relevant

- Formula for IOC Score:

$$IOC = \frac{\Sigma X}{N}$$

The final IOC score for each method was computed as the average of all evaluated questions across the five ITPE categories.

D. Results and Analysis

The following table presents the evaluation results of the three matching methods: SBERT, EmbedRank, and KeyBERT. The average IOC score indicates the overall effectiveness of each approach in aligning job descriptions with relevant exam questions.

Table I: Expert Score of each method

Method	**Direct SBERT**	**PhraseMap**	**KeyAlign**
Database	0.563	0.138	0.425
Network	0.696	0.300	0.716
Cybersecurity	0.458	0.163	0.408
Software	0.617	0.475	0.208
IT Business	0.738	0.371	0.538
Sum	0.614	0.289	0.459

The evaluation of Direct SBERT, PhraseMap, and KeyAlign shows distinct differences in performance (See Table I). **Direct SBERT** achieved the highest accuracy, particularly excelling in IT Business (0.738) and Network (0.696), demonstrating strong semantic alignment. **KeyAlign** performed moderately well, with notable results in Network (0.716) but struggled in Software (0.208). **PhraseMap** had the lowest overall score (0.289), particularly weak in Database (0.138) and Cybersecurity (0.163), reflecting its limitation in capturing contextual meaning. Overall, Direct SBERT proved the most effective, while KeyAlign showed potential, and PhraseMap's reliance on word embeddings limited its accuracy.

V. DISCUSSION

The experiment revealed notable differences in the performance of the three methods. First, **Direct SBERT** outperformed the other methods in terms of contextual understanding, achieving the highest accuracy in matching job descriptions with exam questions. Second, **PhraseMap** performed well when job descriptions contained clear, domain-specific keywords but struggled with more complex descriptions. Third, **KeyAlign** was the fastest method but showed limitations in handling nuanced text that required deeper contextual interpretation.

The superior performance of SBERT can be attributed to its ability to capture contextual relationships beyond simple keyword matching. However, this comes at the cost of higher computational complexity. In contrast, EmbedRank and KeyBERT provide faster results but may not always align well with the intended meaning of job descriptions. The strengths and limitations for each method summarized in Table II.

Table II: Strengths and Limitations Summarized

Method	**Strengths**	**Limitations**
Direct SBERT	High accuracy, strong contextual understanding	Computationally expensive
PhraseMap	Fastest method, easy to implement	Struggles with ambiguous descriptions
KeyAlign	Good for structured text, faster than EmbedRank	Less effective for complex text matching

- These findings have significant implications for automated job-exam matching systems. Future research should consider developing hybrid models that integrate SBERT's contextual understanding with KeyBERT's keyword extraction efficiency to enhance matching accuracy. Expanding datasets to include a wider range of job descriptions would allow for a more comprehensive evaluation of these methods. Additionally, incorporating alternative evaluation metrics, such as Precision-Recall analysis and Mean Reciprocal Rank (MRR), could provide deeper insights into model performance beyond the IOC score.

VI. SUMMARY

This study aimed to develop a method for selecting exam questions based on job description domains by leveraging keyword extraction and semantic similarity techniques. Three methods, KeyAlign, PhraseMap, and Direct SBERT, were employed to match the exams corresponding to the job description contents in different ways. All methods were compared by using 30 job descriptions divided into 5 topics according to ITPE exam contents to evaluate.

Experimental results indicated that all methods effectively identified relevant exam questions, with Direct SBERT providing highest precision in domain-specific contexts. The weighted distribution approach ensured a balanced allocation of questions across different clusters.

However, the study was limited by the dataset size and its focus on ITPE exams, which may not generalize to other domains. Additionally, the imbalance in the number of exam questions across different topics may have influenced the results, as some clusters contained significantly more questions than others. Moreover, the lack of a well-structured exam dataset required manual extraction from files, which introduced the possibility of errors in data processing. Future work could explore hybrid models that integrate keyword-based and deep learning approaches, expand the dataset for broader evaluation, and incorporate additional performance metrics for a more comprehensive analysis.

ACKNOWLEDGMENT

This research has received funding support from the NSRF via the Program Management Unit for Human Resources & Institutional Development, Research and Innovation [grant number B46G670083].

REFERENCES

[1] D. Zhang, J. Liu, H. Zhu, Y. Liu, L. Wang, P. Wang, and H. Xiong, "Job2Vec: Job title benchmarking with collective multi-view representation learning," in *Proc. 28th ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2019. https://doi.org/10.1145/3357384.3357825

[2] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. 2019 Conf. Empirical Methods in Natural Language Processing (EMNLP)*, 2019, pp. 3980–3990. https://doi.org/10.18653/v1/D19-1410

[3] M. Grootendorst, "KeyBERT: Minimal keyword extraction with BERT," Zenodo, 2020. https://doi.org/10.5281/zenodo.4461265

[4] K. Bennani-Smires, C. Musat, A. Hossmann, M. Baeriswyl, and M. Jaggi, "EmbedRank: Unsupervised keyphrase extraction using sentence embeddings," *arXiv preprint arXiv:1801.04470*, 2018. https://doi.org/10.48550/arXiv.1801.04470

[5] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, no. 1, pp. 53–65, 1987.: https://doi.org/10.1016/0377-0427(87)90125-7

[6] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd ed. Elsevier, 2011.

[7] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly Media, 2009.

[8] PyPDF2, "Welcome to PyPDF2," [Online]. Available: https://pypdf2.readthedocs.io/. Accessed: Feb. 13, 2025.