Automating Course Articulation
A Deep Metric Learning Framework Using Public Data

A Thesis submitted to the faculty of

San Francisco State University

In partial satisfaction of the

requirements for

the Degree

Master of Science

in

Data Science and Artificial Intelligence

by

Mark S. Kim

San Francisco, California

May 2025

Certification of Approval

I certify that I have read Automating Course Articulation: A Deep Metric Learning Framework Using Public Data by Mark S. Kim and that in my opinion this work meets the criteria for approving a thesis submitted in partial fulfillment of the requirement for the degree Master of Science at San Francisco State University.

Hui Yang, Ph.D

Professor

Thesis Committee Chair

Arno Puder, Ph.D

Professor

Anagha Kulkarni, Ph.D

Professor

Abstract

The manual process of determining course equivalency is a significant barrier to student mobility and educational equity, leading to substantial credit loss and delayed graduation that disproportionately harms underrepresented students. Previous automated approaches have been limited by a reliance on sensitive student enrollment records or the operational intractability and opacity of using large language models for direct classification.

This thesis introduces and validates a novel framework that overcomes these challenges by decoupling semantic representation from classification, using only publicly available course catalog data. The methodology leverages deep metric learning to fine-tune contextual embedding models on public course text. These specialized embeddings are then used to construct a novel composite distance vector, which serves as a rich feature set for training traditional machine learning classifiers.

Evaluated on a real-world dataset, the proposed framework achieves state-of-the-art accuracy, with F_1-scores exceeding 0.99. The result is a computationally efficient, scalable, and privacy-preserving solution that provides institutions with a practical tool to automate course articulation, reduce administrative burden, and foster a more equitable educational ecosystem.

Acknowledgments

I would like to express my deepest gratitude to my advisors, Professors Hui Yang, Arno Puder, and Anagha Kulkarni. Their patience, support, and confidence in me have been invaluable throughout my graduate journey at SFSU. I also owe a special debt of gratitude to Professor Tao He, whose crucial suggestion to incorporate a global similarity metric with the embedding vectors significantly improved the model's classification performance.

This research was supported by the work of many individuals. I extend my sincere thanks to Natalie Yam, Parth Panchal, and Joanne Park for their assistance with data collection, compilation, and preliminary analysis. I am also grateful to the Program Pathways Mapper (PPM) team, which includes representatives from the Kern Community College District, the Foundation for California Community Colleges, and the California Community Colleges Chancellor's Office, for their partnership and for providing the foundational data for this work.

On a personal note, I am immensely thankful for my dear friends who have been there for me through thick and thin, cheering me on whenever I felt I could not continue. To my good friend, Phil, I offer my deepest appreciation for his incredible generosity and support, which made navigating the financial challenges of graduate school possible. Finally, I am profoundly grateful to Julie and Bita for their invaluable guidance and perspective, which were instrumental to my personal growth and well-being throughout this journey.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

California's public higher education system is a titan of American academia, a complex, three-tiered structure comprising the University of California (UC), California State University (CSU), and the California Community Colleges (CCC) [11]. Collectively, these 149 colleges and universities serve nearly 2.9 million students, forming the largest public higher education system in the United States [11, 27, 7, 6]. A foundational principle of this system is the promise of student mobility, particularly the pathway from a two-year community college to a four-year university. However, the mechanism designed to facilitate this movement, the process of determining course equivalency, or articulation, is a formidable, largely manual process that creates significant barriers for students.

At the heart of this process is the Articulation System Stimulating Interinstitutional Student Transfer (ASSIST), the state's official public repository for articulation agreements [4]. While ASSIST provides a centralized platform for students and advisors to view established equivalencies, it is fundamentally a display for agreements that are negotiated and updated manually by articulation officers at each individual campus [3]. Given the sheer number of institutions, the process of defining and maintaining these agreements is a task of bleak combinatorics, rendering it inefficient, slow, and inherently intractable [22]. This manual paradigm places a considerable burden on academic advisors and administrative staff, who must meticulously review course descriptions and syllabi to compare content, rigor, and learning outcomes [22]. The result is a system that struggles to keep pace with the needs of a vast and mobile student body, making California a critical case study for a problem that extends far beyond its borders.

The challenges exemplified by California's system are a microcosm of a systemic crisis in American higher education. The act of transferring between institutions has become a normative component of the modern student's academic journey. Data from the National Student Clearinghouse Research Center reveals that in the fall of 2023, transfer enrollment constituted 13.2% of all continuing and returning undergraduates [8]. This trend is not static; it represents a post-pandemic resurgence in student mobility, with transfer enrollment growing by 5.3% from Fall 2022 to Fall 2023 and an additional 4.4% in Fall 2024 [8, 9]. This mobile population is increasingly diverse, comprising not only students following traditional two-year to four-year pathways but also a substantial number of returning learners who have previously paused their education. Over half of these returning students opt to re-enroll at a new institution, underscoring the critical role of the transfer system in providing flexible pathways to degree completion [10].

The consequences of this systemic inefficiency are borne almost entirely by the students, manifesting in significant academic and financial setbacks. The most direct and damaging outcome is the loss of earned academic credit. A comprehensive 2017 report by the U.S. Government Accountability Office (GAO) estimated that students who transferred between 2004 and 2009 lost an average of 43% of their credits in the process [32]. This finding is echoed across numerous studies, with reports indicating that more than half of all transfer students lose at least some credits, and approximately one-fifth are forced to repeat courses for which they have already received a passing grade at a previous institution [1].

This loss of credit creates a cascade of negative consequences. It invariably leads to an increased time-to-degree, delaying graduation and entry into the workforce. Each repeated course also carries a financial cost, increasing the total tuition burden and potentially exhausting a student's eligibility for federal financial aid programs like Pell Grants and Direct Loans [32]. A process that is often undertaken to save money (for example, by starting at a less expensive community college) can paradoxically result in a greater overall financial commitment, trapping students in a cycle of additional coursework and debt [5].

The friction and frustration inherent in the transfer process also have a measurable impact on student persistence and graduation. Studies have shown that transfer students, as a group, tend to have lower retention and graduation rates than their peers who begin and end their studies at the same institution [26]. This issue transcends mere administrative inefficiency and becomes a critical matter of educational equity. Low-income students and students from historically underrepresented racial and ethnic groups are more likely to begin their postsecondary journey at community colleges and rely on transfer pathways to attain a bachelor's degree [31]. The recent growth in transfer enrollment has been driven disproportionately by Black and Hispanic students [8]. Therefore, the barriers imposed by an inefficient articulation system such as credit loss, increased cost, and delayed graduation, disproportionately harm the very student populations that institutions are striving to support.

A clear and troubling feedback loop emerges from this analysis. The fundamentally manual and inefficient nature of course articulation is a direct cause of credit loss. This credit loss imposes a tangible academic and financial burden on students which falls most heavily on underrepresented and low-income students, who are a large and growing segment of the transfer population. This disproportionate impact, in turn, undermines institutional goals of improving student retention and closing persistent equity gaps in degree attainment. Thus, the seemingly low-level administrative task of determining course equivalency is revealed to be a significant driver of systemic inequity in higher education. Addressing this challenge through robust automation is not merely an operational optimization; it is a necessary intervention to foster a more equitable, efficient, and supportive educational ecosystem for all students.

1.1 Contributions

This research makes several key contributions to the fields of educational data mining and natural language processing, offering a practical and powerful solution to the long-standing challenge of course articulation.

- **A High-Accuracy, Automated Framework**: This thesis develops and validates a novel framework for determining course equivalency that achieves state-of-the-art accuracy, with F_1-scores exceeding 0.99 on a challenging real-world dataset. Crucially, it accomplishes this using only publicly available course catalog text, making it broadly applicable.

- **An Innovative Feature Engineering Technique**: It introduces a composite distance vector, Δ_c, that uniquely combines element-wise embedding differences with cosine similarity. This technique provides a richer input signal for classification and is shown to demonstrably improve the performance of downstream machine learning models, particularly linear classifiers.

- **A Computationally Efficient and Scalable Approach**: The research demonstrates that by decoupling semantic representation from classification, it is possible to harness the power of deep contextual embeddings without the high computational costs, API dependencies, and opaque nature of direct LLM-based classification. This makes the proposed solution more efficient, scalable, and practical for institutional deployment.

- **A Privacy-Preserving Methodology**: By relying exclusively on public course descriptions, the proposed method circumvents the significant privacy, security, and data access challenges associated with techniques that require sensitive student enrollment records. This makes the framework more ethically sound and generalizable across any pair of institutions, regardless of their data-sharing agreements.

1.2 Thesis Roadmap

The remainder of this thesis is structured to provide a comprehensive account of this research.

- **Chapter 2: Background and Related Work** will provide a detailed in-depth survey of the landscape of student transfer automation and the evolution of technological interventions.
- **Chapter 3: Methodology** will offer a deep dive into the data collection and preparation processes, the specific embedding models evaluated, the construction of the feature vectors, and the theoretical underpinnings of the machine learning classifiers employed.
- **Chapter 4: Experimental Setup and Results** will detail the experimental design, the datasets used for training and validation, and a comprehensive analysis of the classification performance, including ablation studies and model comparisons.

- **Chapter 5: Discussion and Future Work** will interpret the results in a broader context, discuss the limitations of the current study, and outline promising avenues for future research, including the development of a full-scale course recommendation system and the exploration of fine-tuning techniques.
- **Chapter 6: Conclusion** will summarize the key findings of the thesis and reiterate the significance of its contributions to both academic research and the practical administration of higher education.

Chapter 2

Background and Related Works

The manifest inefficiencies and inequities of manual course articulation, exemplified by the challenges within California's vast system, have prompted a range of research efforts aimed at automating the process [20, 25, 21, 17, 33]. These technological interventions have evolved in sophistication, mirroring the broader advancements in natural language processing (NLP) and machine learning. A critical review of this literature reveals a clear trajectory from simple statistical methods to complex deep learning models, with each stage introducing new capabilities while also exposing new limitations. This evolution illuminates the path toward a more robust and scalable solution.

2.1 Keyword and Statistical Methods

The earliest attempts at automating course comparison relied on foundational text analysis techniques that, while computationally simple, lack semantic depth. The most basic systems are essentially search engines or databases that depend on exact keyword matching or pre-populated tables of known equivalencies [18]. These systems are inherently brittle; they cannot recognize semantic variations (e.g., equating "Introduction to Programming" with "Fundamentals of Computer Science I") and require continuous manual updates to remain relevant [29].

A more advanced statistical method, Term Frequency-Inverse Document Frequency (TF-IDF), improves upon keyword matching by vectorizing documents and weighting terms based on their importance. A term's frequency within a single document (TF) is balanced against its rarity across a collection of documents, or corpus (IDF) [2]. This allows the model to assign higher importance to distinctive terms (e.g., "calculus") and lower importance to common words (e.g., "the," "a," "is") [2]. TF-IDF has been a workhorse for information retrieval and has been applied to course similarity tasks [2]. However, the fundamental limitation of TF-IDF and other bag-of-words models is their complete lack of semantic understanding. They treat words as discrete, unrelated tokens and cannot grasp that "calculus" and "differentiation" are related concepts, nor can they distinguish between different meanings of the same word.

2.2 Static Semantic Representations

The development of word embeddings represented the first major leap toward a true semantic understanding of course content. Models like Word2Vec and GloVe, trained on vast text corpora, learn to represent words as dense vectors in a high-dimensional space, where words with similar meanings are positioned closer to one another. For example, the vectors for “car” and “automobile” would be near each other, while being distant from the vector for “planet.” This innovation enabled a more nuanced comparison of texts than was possible with TF-IDF. In the context of educational data mining, these techniques have been applied to content-based course recommendation systems, typically by creating a single vector representation for a course description by averaging the vectors of all its constituent words [23].

Despite this advancement, these models produce static embeddings. Each word is assigned a single, fixed vector regardless of its context. This is a significant drawback, as it fails to account for polysemy: words with multiple meanings. For instance, the word “bank” would have the same vector in the phrases “river bank” and “bank account,” despite their disparate meanings. Furthermore, the common practice of averaging all word vectors to create a document-level representation is a crude heuristic that can dilute or lose critical semantic information, especially in complex or lengthy descriptions.

2.3 Contextual Semantic Representations

The introduction of the transformer architecture, and specifically models like Bidirectional Encoder Representations from Transformers (BERT), revolutionized NLP by enabling the generation of contextual embeddings. In these models, the vector representation of a word is dynamically influenced by the words surrounding it in a sentence [12]. This allows the model to disambiguate word meanings and capture a much richer, more accurate semantic representation of the text. Architectures such as Sentence-BERT (SBERT) were subsequently developed to fine-tune these models specifically for the task of producing semantically meaningful embeddings for entire sentences or short paragraphs, which can then be efficiently compared using metrics like cosine similarity [28].

The most recent evolution in this domain involves the direct application of large-scale generative models, or Large Language Models (LLMs) like GPT-4 and Gemini, for classification tasks. Through sophisticated prompt engineering and in-context learning, these models can be instructed to perform pairwise comparisons of course descriptions and render a judgment on their equivalency. The preliminary work for this thesis explored this very approach, using Google's Gemini Pro to classify course pairs. While these experiments yielded promising accuracy, they also uncovered significant practical and theoretical limitations. The direct use of LLMs for this task is computationally expensive and inefficient, as it requires repeatedly sending full text descriptions to a model API for every comparison. Performance is acutely sensitive to the exact phrasing of the prompt, necessitating a costly and time-consuming iterative tuning process. Most critically, the decision-making process of an LLM is a “black box,” providing a categorical output (e.g. “equivalent”) without a quantifiable similarity score or confidence level. This opacity makes it difficult to rank potential matches, set decision thresholds, or provide transparent justifications for the model’s conclusions. This approach is also ill-suited for handling more complex articulation scenarios, such as one-to-many or many-to-many course mappings.

2.4 Direct LLM Classification

The preliminary work for this thesis directly tested the hypothesis that modern, large-scale generative models could serve as end-to-end classifiers for course equivalency. This investigation used Google’s Gemini Pro model to render judgments on course pairs sourced from public California college catalogs. The experiment compared two conditions: one where the model was provided with the full, raw text of two course descriptions, and another where it was given only a curated list of topics that had been previously extracted from the descriptions by the model itself.

The results were informative. The model achieved a notable 90.5% accuracy when classifying pairs based on their complete, raw text descriptions. This performance was superior to the 81.0% accuracy achieved using pre-extracted topics, suggesting that the model benefited from the full context and that the extraction process, despite careful prompt engineering, resulted in a loss of critical semantic information. A comparative analysis also found that

Gemini Pro outperformed numerous contemporary open-source models in this direct classification task.

However, this exploratory work also revealed significant practical and theoretical limitations that rendered the direct classification approach unsuitable for a robust, scalable solution. The primary drawbacks identified were:

- Computational Inefficiency: The method required repeatedly sending the full text of course descriptions to a model API for every pairwise comparison, a process that is both slow and prohibitively expensive at scale.
- Opacity and Lack of Granularity: The model's output was a categorical decision (e.g., "equivalent") without a quantifiable similarity score or confidence level. This "black box" nature prevents the ranking of potential matches, the setting of nuanced decision thresholds, and the ability to provide transparent justifications for its conclusions.
- Extreme Prompt Sensitivity: The model's performance was acutely sensitive to the exact phrasing of the instructional prompt, necessitating a costly and time-consuming iterative tuning process that was not generalizable across different models.
- Insufficient Accuracy for Production Use: While an accuracy of 90.5% is admirable for a preliminary experiment, it corresponds to an error rate of nearly one in ten decisions. Given the significant negative consequences that errors can inflict on student—including lost credits, increased financial burden, and delayed graduation—this level

of reliability is insufficient for a production system. The framework ultimately developed in this thesis targets and achieves accuracy exceeding 99%, a standard more appropriate for the high-stakes nature of student articulation.

These findings were pivotal. While the experiment confirmed the profound semantic understanding of modern LLMs, it simultaneously underscored the operational intractability of using them as direct classifiers. The inefficiency, opacity, and brittleness of the approach made it clear that a more sophisticated framework was needed. This conclusion was the direct impetus for the methodology proposed in this thesis, which decouples semantic representation from classification to create a system that is not only highly accurate but also efficient, transparent, and scalable.

2.5 Enrollment-Based Approaches

It is essential to situate this research within the context of parallel efforts that leverage different data sources. A notable body of work has demonstrated that course similarity and prerequisite relationships can be predicted with high accuracy by analyzing student enrollment data. Models such as *course2vec* learn course embeddings not from their textual descriptions, but from the patterns of which courses students tend to take together [24]. The underlying principle is that courses frequently taken in the same semester or in sequence likely share a functional or topical relationship [24].

While this behavioral approach is powerful, its reliance on large-scale, proprietary institutional datasets of student records presents two major obstacles. First, it raises significant data privacy and security concerns, as it involves the analysis of sensitive student information [30]. Second, it limits the scalability and generalizability of the solution. The model is only applicable at institutions that can provide access to such data, and it cannot be used to compare courses between two institutions that have no history of student transfer between them. This approach is therefore not a universal solution for the broader course articulation problem.

The evolution of these varied approaches reveals a fundamental trade-off: as models gain greater semantic power, they tend to become more computationally intensive, less interpretable, and more demanding of specialized or private data. The limitations of direct LLM classification (cost, opacity) and enrollment-based methods (data privacy, limited access) point toward the need for a new paradigm. An ideal solution should harness the semantic power of large pre-trained models without inheriting their operational burdens. This suggests that the next logical step is not simply a larger, more complex end-to-end model, but rather a more intelligent, hybrid framework. Such a framework would decouple the task of deep semantic representation from the task of final classification, allowing each component to be optimized for what it does best. This conceptual shift forms the central motivation for the methodology proposed in this thesis. Table 2.1 summarizes the primary methods for determining course transferability.

Table 2.1: Comparative Taxonomy of Course Equivalency Determination Methods

Approach	Key Characteristics	Data Source(s)	Semantic Capability	Strengths	Limitations
Manual Review	Human experts (advisors, faculty) compare syllabi descriptions.	Course Catalogs, Syllabi	High (Human-level)	Nuanced, context-aware, trusted by faculty.	Extremely slow, not scalable, subjective, prone to inconsistency.
Keyword/ TF-IDF	Bag-of-words representation, statistical term weighting.	Course Catalogs	None to Low	Simple, computationally cheap, easy to implement.	Fails to capture synonyms, context, or true semantic meaning
Static Embeddings (Word2Vec/ GloVe)	Pre-trained word vectors, often averaged for document representation.	Course Catalogs	Medium	Captures word-level semantics, better than TF-IDF.	Context-insensitive, averaging vectors loses information.
Enrollment-Based (e.g., course2vec)	Embeddings learned from student co-enrollment patterns.	Proprietary Student Records	High (Behavioral)	Captures functional relationships between courses, highly predictive.	Requires access to sensitive private data, not generalizable, privacy concerns.
Direct LLM Classification	End-to-end classification using prompt engineering.	Course Catalogs	Very High	High accuracy potential, understands complex language.	Computationally expensive, “black box” opacity, prompt sensitive, no quantifiable similarity score, risk of hallucinations.
Proposed Method (Embeddings + ML)	Deep contextual embeddings as features for traditional classifiers.	Course Catalogs	Very High	State-of-the-art accuracy, computationally efficient, quantifiable, uses public data only.	Relies on the quality of the pre-trained embedding model.

Chapter 3

Methodology

In response to the limitations of existing methods, this thesis proposes and validates a novel framework for automating course equivalency determination. This approach is designed to achieve state-of-the-art accuracy while remaining computationally efficient, transparent, and reliant only on publicly available data. It achieves this by strategically decoupling the process of semantic understanding from the final classification task, leveraging the distinct strengths of deep embedding models and traditional machine learning classifiers.

The foundational principle of the proposed framework is to utilize state-of-the-art deep embedding models not as end-to-end classifiers, but as highly sophisticated feature extraction engines. In this paradigm, a publicly available course description, in its raw text form, is processed by a pre-trained contextual embedding model. The model's task is to convert the unstructured text into a high-dimensional numerical vector that encapsulates the rich semantic content of the course.

This computation is performed once for each course in the catalog, transforming the entire corpus of unstructured text into a structured database of semantic vectors. This pre-processing step creates a reusable and efficient representation of each course, directly addressing the primary drawbacks of the direct LLM classification approach. By storing these embeddings, the framework eliminates the need to repeatedly process the same raw text through a costly API, drastically reducing computational overhead and latency for subsequent comparisons. This makes the system inherently more scalable and suitable for real-world applications that may involve hundreds of thousands of courses.

This chapter delineates the comprehensive and systematic methodology employed to develop and evaluate a sophisticated classification pipeline. The central objective of this research is to construct a robust predictive model capable of accurately classifying domain-specific textual data. The methodological approach is predicated in a multi-stage process, designed to address the inherent complexities of natural language and to maximize predictive performance. This process begins with the acquisition and preparation of the data corpus, proceeds to the creation of semantically rich numerical representations through deep metric learning—a process that uses a deep neural network to learn an embedding space where the distance (or metric) between similar items is minimized and the distance between dissimilar items is maximized—and culminates in the systematic training and evaluation of a diverse suite of machine learning classifiers.

The narrative of this chapter follows the logical and sequential flow of the research execution. First, it details the data acquisition, pre-processing, and structuring protocols, with a particular focus on the preparation of data for advanced metric learning techniques. Second, it provides a deep dive into the core of the feature engineering strategy: the fine-tuning of a pre-trained transformer model using a triplet loss function. This section offers a rigorous examination of the underlying architecture, the theoretical principles of the chosen loss function, and the rationale behind the selection of optimizer and learning rate scheduling framework. Third, the chapter describes the process of generating the feature vectors from fine-tuned models, effectively transforming the unstructured text problem into a structured format suitable for classical machine learning. Fourth, it presents the theoretical foundations and application of each downstream classification algorithm evaluated in this study, justifying their inclusion based on their distinct learning paradigms. Finally, the chapter outlines the dual evaluation framework used to select the optimal embedding model during the fine-tuning phase, and assess the performance of the final classifiers on a held-out test set. This structured and principled methodology ensures the rigor, reproducibility, and validity of the research findings presented in subsequent chapters.

3.1 Data Corpus and Pre-processing

The foundation of any machine learning endeavor is the data upon which it is trained and evaluated. This section describes the characteristics of the dataset used in this study, the pre-processing pipeline applied to ensure data quality and consistency, and the specific data structuring techniques required to facilitate the deep metric learning phase of the research.

Dataset Characterization

The primary data for this research was sourced from an expanded dataset provided in partnership with Program Pathways Mapper (PPM), initially consisting of 2217 labeled courses. The class label for each course is its Course Identification Numbering System (C-ID) code, which serves as a ground truth for equivalency.

A crucial pre-processing step was performed to ensure that sufficient examples existed for robust training and evaluation. To guarantee that both the training and test sets would contain at least one pair of equivalent courses for every class, all classes with fewer than four courses in the total dataset were removed. This filtering process resulted in a final corpus of 2157 courses distributed across 157 distinct C-ID classes.

To ensure a rigorous and unbiased evaluation of the final models, this final dataset was partitioned into two distinct, non-overlapping subsets: a training set and a test set. A stratified 50/50 split was employed, using the C-ID code as the stratification key. This resulted in a training set of 1078 courses and a test set of 1079 courses. The stratification ensures that each of the 157 classes is represented in both subsets, with each class containing between 2 and 33 equivalent courses per set. The test set is held in reserve and used only once for the final, conclusive evaluation of the optimized classification pipeline, providing an honest estimate of the model's generalization performance on unseen data.

The training set serves a dual purpose in this methodology, a strategy made possible by the different data formats required by the training and validation procedures. For the fine-tuning process itself, the training set is used to generate (anchor, positive, negative) triplets as required by the Triplet Loss functions. For the validation within the fine-tuning process, this same training set is utilized to generate binary pairs of “equivalent” and “non-equivalent” course pairs. This validation set is then used with the BinaryClassificationEvaluator to monitor model performance during training and select the best model checkpoint. This approach allows for robust validation without data leakage, as the model is evaluated on our primary task (binary pair classification) than the one it is directly optimizing for (triplet distance minimization).

Document Normalization for Embedding

In a departure from conventional NLP pipelines, this research deliberately eschewed standard text pre-processing techniques such as lowercasing, stop-word removal, or the stripping of special characters. This decision was made to more accurately simulate a real-world use case where input data, such as course information copied directly from a university website, may be imperfectly formatted or contain extraneous characters. The methodology, therefore, relies on the inherent semantic power and robustness of modern transformer-based embedding models to interpret and handle this ”raw” text.

Instead of cleaning the text, a normalization step was performed to create a consistent, structured input for the embedding models. A new field, “Formatted Course Info,” was generated for each course by concatenating four key pieces of information: the department name, the department course number, the course title, and the full course description. This concatenated string serves as the single document representation for each course and is the direct input for the document embedding process described in Section

Data Structuring for Fine-Tuning

The family of triplet loss functions operates on a conceptual structure of (anchor, positive, negative) triplets [28, 14]. However, a key advantage of the batch-based triplet loss functions used in this research is that they do not require the manual, pre-generation of these triplets. Instead, the model is trained on a dataset of (text, label) pairs [14]. The loss function then intelligently forms the necessary triplets on-the-fly from within each mini-batch of data. This online mining approach is significantly more efficient and effective than offline strategies. These triplets are defined as:

- **Anchor (A)**: A reference sentence or text sample.
- **Positive (P)**: A sentence that belongs to the same class as the anchor.
- **Negative (N)**: A sentence that belongs to a different class from the anchor.

The training process involves feeding these triplets to the model and optimizing it to minimize the distance between the anchor and positive embeddings while maximizing the distance between the anchor and negative embeddings in the vector space.

The generation of these triplets from the normalized, labeled training set is a critical preparatory step. A crucial constraint imposed by triplet loss functions, particularly those available in the sentence-transformers library, is that the training data must contain a minimum of two examples for each class label [15]. This is not a superficial requirement but a fundamental necessity for the loss computation to be valid. The formation of an (anchor, positive) pair requires selecting two distinct samples from the same class [15]. If a class is represented by only one sample, it can never form a positive pair and can only contribute to (anchor, negative) pairings.

This constraint has a direct and significant implication for the data loading and batching strategy. A simple random sampling approach could easily result in mini-batches that lack the necessary class diversity, containing only one or even zero instances of certain classes. In such a scenario, no valid triplets could be formed for the anchors of those underrepresented classes within that batch, leading to an inefficient and potentially biased training process. To mitigate this, a more sophisticated data loading mechanism was required. Rather than implementing a fully custom solution, this research leveraged a specialized, built-in feature of the sentence-transformers library: the `GroupByLabelBatchSampler`. This sampler, activated by setting the `batch_sampler` argument to `BatchSamplers.GROUP_BY_LABEL` during trainer configuration, is specifically designed for use with triplet loss functions [16]. It addresses the sampling challenge by ensuring that each mini-batch is constructed by grouping samples with the same label, thereby guaranteeing that every batch contains the necessary class diversity to form valid and informative triplets for all anchors. This demonstrates a key principle of the methodology: the data preparation and loading stage is not an independent preliminary step but is intrinsically linked to and constrained by the fine-tuning objective.

3.2 Embedding Model Selection and Fine-Tuning

A central hypothesis of this research is that a generic, pre-trained language model can be adapted to produce highly specialized and semantically rich embeddings for the specific domain of the data corpus. These bespoke embeddings are expected to provide a more discriminative feature representation for downstream classification tasks compared to off-the-shelf embeddings. This section details the architecture, learning objective, and training protocol used to achieve this adaptation through a process of deep metric learning.

Selection

The selection of an appropriate embedding model is a critical first step that influences the entire downstream pipeline. Rather than selecting a single model, this research began with a broad preliminary analysis of a variety of open-source embedding models to identify strong candidates for a more in-depth, comparative study. The initial models reviewed spanned a wide range of parameter sizes and characteristics which are summarized in Table 3.1.

This preliminary evaluation was conducted using a simple accuracy metric: for a given anchor course, a model was considered correct if the cosine similarity to an equivalent course was greater than the similarity to a non-equivalent course. This initial screening revealed

Table 3.1: Initial Embedding Model Review

Model Name	Rank*	Params†	Dims	Acc
GIST-small-Embedding-v0	41	33	384	0.9759
bge-small-en-v1.5	47	33	384	0.9670
GIST-Embedding-v0	33	109	768	0.9768
bge-base-en-v1.5	35	109	768	0.9732
gte-base-en-v1.5	31	137	768	0.9732
mxbai-embed-large-v1	24	335	1024	0.9759
gte-large-en-v1.5	21	434	1024	0.9777
multilingual-e5-large-inst	34	560	514	0.9670
stella_en_1.5B_v5	3	1543	8192	0.9857
SFR-Embedding-2_R	4	7111	4096	**0.9839**
Agte-Qwen2-7B-instruct	5	7613	3584	0.9804
nvidia/NV-Embed-v2	1	7851	4096	0.9831

* Huggingface Overall Leaderboard Rank

† in Millions

that while performance varied, many models achieved high accuracy, with a sample mean of 0.9767 and a standard deviation of 0.00605. Based on these results and a desire to evaluate a representative spectrum of model sizes, three models were selected for the primary analysis:

- `BAAI/bge-small-en-v1.5` (BGE): Representing a high-performing small model (33M parameters).
- `avsolatorio/GIST-Embedding-v0` (GIST): Representing a medium-sized model (109M parameters).
- `nvidia/NV-Embed-v2` (NVE): Representing a large-scale model (7.85B parameters).

At a later stage of the research, `Salesforce/SFR-Embedding-2_R` (SFR) was also included for additional comparison due to its strong performance on public leaderboards. The foundational architecture for all these models is the transformer, which allows them to generate rich, contextual embeddings. By employing a pooling layer (typically mean-pooling), these models produce a single, fixed-size vector for each input document, making them highly suitable for feature extraction.

Metric Learning with Batch Triplet Loss Functions

To determine if performance could be further improved, two of the selected models, BGE and GIST, were subjected to a fine-tuning process using a metric learning approach. Metric learning aims to learn an embedding space where the geometric distance between samples corresponds to their semantic similarity.

Theoretical Foundation of Triplet Loss

The concept of Triplet Loss was first introduced by Yu, et al. in the context of face recognition and has since been widely applied to supervised similarity learning [34]. It operates on the (Anchor, Positive, Negative) triplets described in Section 3.1. The fundamental goal of Triplet Loss is to train the embedding function, $f(x)$, to map inputs into a vector space where the distance between an anchor sample (A) and a positive sample (P) from the same class is smaller than the distance between the anchor and a negative sample (N) from a different class. To prevent the model from collapsing all embeddings to a single point and to ensure a meaningful separation, the loss function enforces a margin, α. The distance between the anchor-positive pair must be smaller than the distance to the anchor-negative pair by at least this margin.

The mathematical formulation of the Triplet Loss is given by:

$$L(A, P, N) = \max(d(f(A), f(P)) - d(f(A), f(N)) + \alpha, 0).$$

Here, d represents a distance metric, which in this study is the Euclidean distance. The $\max(d(f(A), f(P)) - d(f(A), f(N)) + \alpha, 0)$ component ensures that the loss is only incurred for triplets that violate the margin constraint. If the negative sample is already further from the anchor than the positive sample by at least the margin, the loss for that triplet is zero, and no weight update is performed.

Online Triplet Mining and Batch-based Losses

A naive implementation of Triplet Loss would form all possible triplets from the training data, an approach known as ”offline” mining. This is computationally infeasible for large datasets and highly inefficient, as the vast majority of triplets are ”easy” and provide no useful learning signal. A more effective approach is ”online” mining, where informative triplets are selected on-the-fly from within each mini-batch of training data. Within this paradigm, triplets can be categorized by their difficulty :

- **Easy Triplets**: These triplets already satisfy the margin constraint, i.e., $d(A, P)+\alpha < d(A, N)$. They result in zero loss and do not contribute to learning.

- **Semi-Hard Triplets**: In these triplets, the negative sample is more distant than the positive sample but still lies within the margin, i.e., $d(A, P) < d(A, N) < d(A, P) + \alpha$. These triplets have a positive loss and provide a useful, stable gradient for training.

- **Hard Triplets**: These are the most challenging cases, where the negative sample is closer to the anchor than the positive sample, i.e., $d(A, N) < d(A, P)$. These triplets produce the largest loss values and provide the strongest learning signal, forcing the model to learn fine-grained distinctions.

Empirical Evaluation of Batch Triplet Loss Functions

Recognizing that different triplet mining strategies can significantly impact model performance, this research empirically evaluated all four primary batch-based triplet loss implementations available in the sentence-transformers library to identify the optimal strategy for this specific dataset and task. This approach ensures that the chosen loss function is best suited to the data's characteristics, rather than relying on a single, pre-selected method. The evaluated loss functions were:

- **BatchAllTripletLoss**: This function computes the loss for all valid triplets that can be formed within a given batch. While this approach is comprehensive, it can be computationally intensive, and the learning signal can be diluted by the high number of "easy" triplets that contribute zero to the loss.

- **BatchSemiHardTripletLoss**: This function focuses the training effort by considering only semi-hard triplets for each anchor. This is a common strategy that provides stable training but can sometimes lead to slower convergence as it ignores the most challenging ”hard” examples that often provide the strongest learning signal.

- **BatchHardTripletLoss**: This function implements a more aggressive mining strategy. For each anchor sample in a mini-batch, it identifies the hardest positive (the positive sample that is furthest away) and the hardest negative (the negative sample that is closest). The loss is then computed on these most challenging pairs. This can accelerate convergence but is also known to be “temperamental,” potentially leading to a noisy and difficult optimization landscape.

- **BatchHardSoftMarginTripletLoss**: This function is a variation of BatchHardTripletLoss that also focuses on the hardest positive and negative samples but does not require a manually specified margin parameter. This simplifies hyperparameter tuning while still leveraging an aggressive mining strategy.

The final model used for feature generation was trained with the loss function that demonstrated the best performance on the validation set during this empirical evaluation.

Optimization and Training Protocol

The successful fine-tuning of a model with a challenging objective like `BatchHardTripletLoss` is critically dependent on the choice of optimizer and learning rate schedule. The components selected for this research, AdamW and Cosine

Annealing with Warm Restarts, were not chosen in isolation but as parts of a cohesive, synergistic framework designed to ensure stable and effective learning.

Optimizer Selection: AdamW

The optimization of the model's weights was performed using the AdamW optimizer. The foundation of AdamW is the Adam (Adaptive Moment Estimation) optimizer, which has become a de facto standard in deep learning due to its computational efficiency and effectiveness. Adam combines the benefits of two other optimization algorithms: the momentum method, which helps accelerate gradient descent in the relevant direction, and RMSprop, which adapts the learning rate for each parameter based on the magnitude of past gradients. This per-parameter adaptive learning rate makes Adam particularly well-suited for the noisy gradients that can arise from `BatchHardTripletLoss`.

However, the standard implementation of Adam has a subtle flaw in how it handles L2 regularization, also known as weight decay. In standard Adam, the weight decay term is coupled with the gradient update itself. This means that the regularization effect is influenced by the magnitude of the gradient, leading to an inconsistent application of weight decay. For parameters with large gradients, the effective weight decay is reduced, which can hinder proper regularization and lead to poorer generalization [19].

AdamW, proposed by Loshchilov and Hutter, corrects this flaw by decoupling the weight decay from the gradient update step. The adaptive moment updates are performed as in standard Adam, but the weight decay is applied as a separate, final step, directly modifying the weights. This decoupled approach ensures that weight decay acts as a true regularizer, penalizing large weights uniformly without interfering with the optimizer's adaptive learning mechanism. For a complex and potentially unstable training process like the one employed here, the enhanced generalization and training stability offered by AdamW make it a superior choice over standard Adam [19]. It provides the necessary regularization to prevent overfitting on the "hard" examples mined by the loss function, while preserving the adaptive learning that helps navigate the complex loss surface.

Learning Rate Schedule: Cosine Annealing with Warm Restarts

Static or linear learning rates are often suboptimal for training deep neural networks. A global learning rate schedule, which dynamically adjusts the learning rate during training, can significantly improve convergence and final model performance, even when using an adaptive gradient algorithm such as AdamW [19]. For this research, the `CosineAnnealingWarmRestarts` schedule was employed. This schedule combines two powerful concepts: cosine annealing and warm restarts. Cosine annealing smoothly decays the learning rate from an initial maximum value, ν_{max}, to a minimum value, ν_{min}, following the shape of a cosine curve. The learning rate ν_t at a given epoch T_{cur} within a cycle of length T_i is calculated as:

$$\nu_t = \nu_{\text{min}} + \frac{1}{2}\left(\nu_{\text{max}} - \nu_{\text{min}}\right)\left(1 + \cos\left(\frac{T_{\text{cur}}}{T_{\text{max}}}\pi\right)\right)$$

This smooth, gradual decay is often more effective than abrupt step-wise decays, allowing the model to fine-tune its parameters more carefully as it approaches a minimum in the loss landscape [13]. The warm restarts technique introduces periodic "restarts" into the training process. After a specified number of epochs, T_0, the learning rate is abruptly reset to its initial maximum value, ν_{max}, and the cosine decay cycle begins anew. The length of subsequent cycles can be progressively increased by a multiplicative factor, T_{mult}. For example, if $T_0 = 10$ and $T_{\text{mult}} = 2$, the restarts would occur after 10, 30 (10+20), 70 (10+20+40), and so on, epochs.

The combination of these two techniques forms a powerful optimization strategy. The aggressive nature of `BatchHardTripletLoss` creates a complex loss surface with many sharp, suboptimal local minima. A standard decay schedule might cause the optimizer to converge into one of these minima and become stuck. The "warm restarts" provide a crucial escape mechanism. When the model's performance on the validation set begins to plateau, suggesting convergence into a local minimum, the sudden increase in the learning rate effectively "kicks" the optimizer out of that basin, allowing it to explore other regions of the loss landscape. The subsequent smooth cosine decay then enables the optimizer to carefully descend into any new, potentially broader and deeper, minimum it discovers.

This entire fine-tuning framework (`BatchHardTripletLoss`, `AdamW`, and `CosineAnnealingWarmRestarts`) represents a carefully orchestrated system. The loss function provides a powerful, albeit challenging, learning signal. The optimizer provides a stable, well-regularized mechanism for applying weight updates based on that signal. The learning rate schedule provides a global exploration strategy to guide the optimizer across the complex landscape, preventing premature convergence and increasing the likelihood of finding a high-quality solution. This synergistic design is critical for successfully training a highly discriminative embedding model under these demanding conditions.

3.3 Generating Course Embeddings

Upon completion of the model selection and fine-tuning processes, the next stage of the methodology involves using the selected models to transform the entire text corpus into structured numerical formats suitable for a comparative analysis of downstream machine learning classifiers.

This process was executed by loading each selected embedding model and operating it in inference mode. The "Formatted Course Info" text for each course in the training and test sets was individually fed into each model to generate a fixed length embedding vector. This procedure was performed for the models summarized in Table 3.2.

Table 3.2: Embedding Models & PCA Explained Variance

		# of Parameters	Embedding	Explained Variance (# of PCs)		
Model Name	Type Used*	(in Millions)	Dimensions	70%	80%	90%
BAAI/bge-small-en-v1.5	OTS & FT	33	384	28	45	76
avsolatorio/GIST-Embedding-v0	OTS & FT	109	768	23	40	73
nvidia/NV-Embed-v2	OTS	7851	4096	20	37	73
Salesforce/SFR-Embedding-2_R	OTS	7111	4096	20	37	73

* OTS: Off-The-Shelf; FT: Fine-Tuned

The outcome of this procedure is the creation of multiple distinct sets of numerical matrices (one for the training set and one for the test set per embedding model). In each matrix, every row corresponds to a specific text sample from the original corpus, and the columns represent the dimensions of that model's embedding space. These matrices, along with their corresponding class labels, constitute the final feature sets that serve as the direct input for the suite of classification models evaluated in the next section. This step marks the critical transition from the unstructured domain of natural language processing to the structured domain of tabular data analysis, upon which the final classification experiments are built.

3.4 Feature Vector Construction

The high-dimensional embedding vectors generated in the previous step, while semantically rich, are not the final features used for classification. To prepare the data for the downstream classifiers, a two-stage feature engineering pipeline was executed. This pipeline first applies various dimensionality reduction techniques to the embeddings and then constructs pairwise difference vectors from these reduced (and original) embeddings to explicitly represent the relationship between two courses for the equivalency classification task.

Dimensionality Reduction

High-dimensional data can present challenges for some machine learning algorithms, a phenomenon often referred to as the "curse of dimensionality." To investigate the impact of dimensionality on classifier performance and to potentially create more robust feature sets, a systematic process of dimensionality reduction was applied to the embeddings generated by each model.

This process was managed by a dedicated `EmbeddingReducer` class, which applied several reduction techniques to the training and test sets. To ensure the integrity of the evaluation and prevent data leakage, the reducer models were fit exclusively on the training data. The same fitted models were then used to transform the test data. This methodology guarantees that no information from the test set influences the parameters of the reduction models. This step resulted in multiple versions of the training and test embedding matrices: the original high-dimensional version, and several new versions corresponding to each reduction technique applied and the number of dimensions targeted. A PCA analysis was employed to identify the number of principal components that explained 70%, 80%, and 90% of the variance for each model. These values and a static 4 and 7 dimensions were used to generate a wide variety of vectors of varying dimensions to train with and test.

Global and Local Distance Vector

The ultimate goal of this research is to classify pairs of courses as either “equivalent” or “not equivalent.” This requires input features that represent the relationship between two courses, not just the characteristics of a single one. Our preliminary analysis showed that relying on a single, holistic metric like cosine similarity was insufficient for establishing a clear and reliable decision boundary. To overcome this, we designed a composite feature vector that provides a richer, more discriminative representation of a course pair's relationship.

The feature vector, denoted as Δ_c, is constructed by concatenating the element-wise difference of the two course embedding vectors ($\mathbf{A}$ and $\mathbf{B}$) with their cosine similarity. The formal definition is as follows:

$$\Delta_c = \left(a_1 - b_1, \ldots, a_k - b_k, \frac{\mathbf{A} \cdot \mathbf{B}}{\| \mathbf{A} \| \| \mathbf{B} \|} \right)$$

where $\mathbf{A} = (a_1, \ldots, a_k)$ and $\mathbf{B} = (b_1, \ldots, b_k)$ are the k-dimensional embedding vectors for the two courses. This design is powerful because it provides the subsequent classifier with two distinct types of information simultaneously. The element-wise difference captures granular, dimension-specific (local) disparities between the two semantic representations, while the cosine similarity provides a single, normalized measure of their overall (global) alignment in the vector space. This composite vector creates a much more informative and discriminative feature set than a single similarity score alone.

To generate the data for our models, a `CoursePairGenerator` class was implemented. This utility systematically created the training and test datasets by first assembling positive pairs (courses with the same C-ID label) and negative pairs (courses with different C-ID labels). For each of these pairs, it then computed the composite feature vector Δ_c, resulting in a highly informative and discriminative feature set ready for classification.

This procedure was applied to every set of embeddings (e.g., BGE-original, BGE-PCA4-reduced, GIST-original, etc.), resulting in a comprehensive suite of training and testing datasets. Each dataset corresponds to a unique combination of an embedding model and a dimensionality reduction technique, ready for evaluation by the downstream classifiers detailed in the next section.

3.5 Classification Models

To determine the most effective method for classifying the generated pairwise feature vectors, a broad suite of machine learning algorithms was evaluated. This process began with a comprehensive initial evaluation of eight different models to understand which algorithmic families were best suited to the data. Based on those preliminary results, a smaller, more focused set of high-performing models was selected for the final, in-depth analysis.

Initial Model Evaluation

The initial evaluation included a diverse set of classifiers, each chosen to test a different hypothesis about the structure of the feature space:

- **Linear Models (Logistic Regression, Ridge, Lasso)**: These models were used to establish a baseline and determine the degree of linear separability of the data. Their simplicity and interpretability make them excellent for understanding the foundational difficulty of the classification task.

- **Instance-Based Model (k-Nearest Neighbors)**: KNN was included to probe the local structure of the feature space. Its performance indicates whether courses with the same equivalency status form dense, localized clusters.

- **Kernel-Based Model (Support Vector Machine)**: An SVM with a non-linear kernel was used to test for complex, non-linear decision boundaries that linear models cannot capture.

- **Ensemble Model (Random Forest)**: Random Forest was chosen for its robustness to overfitting and its ability to capture complex feature interactions by combining the predictions of many decision trees.

- **Probabilistic Models (LDA and QDA)**: Linear and Quadratic Discriminant Analysis were used to test assumptions about the geometric distribution of the data, specifically whether the classes share a common covariance (LDA) or have unique ones (QDA).

Final Classifier Selection

Based on the preliminary results from the comprehensive evaluation, four models were selected for the final analysis due to their consistently strong performance: K-Nearest Neighbors (KNN), Support Vector Machine (SVM), Random Forest (RF), and XGBoost. XGBoost, a powerful gradient boosting implementation, was added at this stage to include a state-of-the-art boosting algorithm known for its high performance on structured data.

These models represent the most promising approaches for this classification task, covering instance-based, maximal-margin, and advanced ensemble methods.

3.6 Multi-Stage Evaluation

A rigorous, multi-stage evaluation framework was designed to systematically narrow down the optimal combination of embedding models, dimensionality reduction techniques, and classifiers.

Stage 1: Preliminary Embedding Model Selection

The process began with a broad analysis of twelve open-source embedding models to identify strong candidates for more intensive study. This initial screening was performed on a small, manually curated dataset. The evaluation metric was a simple accuracy score based on cosine similarity: for a given course, a model was scored as correct if an equivalent course had a higher cosine similarity than a non-equivalent course. This efficient, low-cost evaluation allowed for the rapid elimination of poorly performing models, resulting in the selection of three top candidates representing a range of sizes: BAAI/bge-small-en-v1.5 (BGE), avsolatorio/GIST-Embedding-v0 (GIST), and nvidia/NV-Embed-v2 (NVE).

Stage 2: Comprehensive Classifier and Reducer Evaluation

Using the three selected embedding models and the initial small dataset, a comprehensive GridSearchCV was conducted. This stage systematically tested each embedding model in combination with multiple dimensionality reduction techniques (PCA, t-SNE, and PaCMAP at 4 and 7 dimensions, as well as with 70%, 80%, and 90% explained variance for PCA) and the full suite of eight initial classifiers (Logistic Regression, Ridge, Lasso, KNN, SVM, Random Forest, LDA, and QDA). The goal of this exhaustive search was to identify the most effective downstream classifiers and to understand the impact of dimensionality reduction on performance.

Stage 3: Evaluation During Fine-Tuning

With the larger, more robust PPM dataset, the two most promising non-proprietary models (BAAI/bge-small-en-v1.5 and avsolatorio/GIST-Embedding-v0) were fine-tuned to specialize them for the course equivalency task. To monitor the quality of the learned embeddings during this process, the `BinaryClassificationEvaluator` from the sentence-transformers library was employed. At the end of each training epoch, the evaluator was run on binary-labeled course pairs generated from the training set. The primary metric monitored was Average Precision based on cosine similarity, and the model checkpoint that achieved the highest score on these validation pairs was saved as the best model for that fine-tuning run.

Stage 4: Final Downstream Classifier Evaluation

The final and definitive evaluation was conducted on the held-out test portion of the PPM dataset. This stage used the feature vectors generated from the best-performing embedding models (both the original off-the-shelf versions and the newly fine-tuned versions). These feature sets were then used to train and evaluate the final selection of high-performing classifiers: KNN, Random Forest, SVM, and XGBoost. This ensures an unbiased assessment of the complete pipeline's ability to generalize to new, unseen data. To provide a comprehensive view of model performance, the standard suite of classification metrics was calculated from the confusion matrix: accuracy, precision, recall, and F_1-Score.

Beyond classification accuracy, the computational efficiency of each model pipeline was assessed by measuring both training and inference times from the hyperparameter cross-validation grid search. While training time provides insight into the resources required to develop a model, inference time is the more critical metric for this research's use case. Inference time directly affects the system's responsiveness and scalability in a production environment, making it a key factor in determining the practical viability of a given solution.

The performance of each of the final classifiers on each of the final feature sets was measured using these metrics. The results of this final evaluation, which form the core findings of this thesis, will be presented and analyzed in the subsequent chapter.

Bibliography

[1] Public Agenda. *Beyond Transfer: Insights from a Survey of American Adults.* https://publicagenda.org/resource/beyond-transfer/ (visited on 06/30/2025).

[2] Akiko Aizawa. "An information-theoretic perspective of tf-idf measures". In: *Information Processing & Management* 39.1 (2003), pp. 45–65. ISSN: 0306-4573. DOI: https://doi.org/10.1016/S0306-4573(02)00021-3. https://www.sciencedirect.com/science/article/pii/S0306457302000213.

[3] ASSIST. *Frequently Asked Questions.* https://resource.assist.org/FAQ (visited on 06/30/2025).

[4] ASSIST. *General Information.* https://resource.assist.org/About/General-Information (visited on 06/30/2025).

[5] Leticia Tomas Bustillos et al. *The Transfer Maze: The High Cost to Students and the State of California.* The Campaign for College Opportunity, Sept. 17, 2017.

[6] California Community Colleges Chancellor's Office. *Management Information Systems Data Mart.* 2024. `https://datamart.cccco.edu/Students/Student%5C_Headcount%5C_Term%5C_Annual.aspx` (visited on 09/13/2024).

[7] California State University Office of the Chancellor. *Enrollment.* 2024. `https://www.calstate.edu/csu-system/about-the-csu/facts-about-the-csu/enrollment` (visited on 09/13/2024).

[8] National Student Clearinghouse. *College Transfer Enrollment Grew by 5.3% in the Fall of 2023.* `https://www.studentclearinghouse.org/news/college-transfer-enrollment-grew-by-5-3-in-the-fall-of-2023/` (visited on 06/30/2025).

[9] National Student Clearinghouse. *College Transfer Enrollment Grew for Third Straight Year.* `https://www.studentclearinghouse.org/news/college-transfer-enrollment-grew-for-third-straight-year/` (visited on 06/30/2025).

[10] National Student Clearinghouse. *DATA DIVE: Returning Learners Lead Transfer Population.* `https://www.studentclearinghouse.org/nscblog/data-dive-returning-learners-lead-transfer-pop/` (visited on 06/30/2025).

[11] Kevin Cook. *California's Higher Education System.* 2024. `https://www.ppic.org/publication/californias-higher-education-system/` (visited on 09/06/2024).

[12] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.* 2019. arXiv: `1810.04805 [cs.CL]`. `https://arxiv.org/abs/1810.04805`.

[13] The Pytorch Foundation. *CosineAnnealingLR.* `https://docs.pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.CosineAnnealingLR.html` (visited on 06/30/2025).

[14] Inc. Hugging Face. *Loss Overview.* `https://sbert.net/docs/sentence_transformer/loss_overview.html` (visited on 06/30/2025).

[15] Inc. Hugging Face. *Losses.* `https://sbert.net/docs/package_reference/sentence_transformer/losses.html` (visited on 06/30/2025).

[16] Inc. Hugging Face. *Samplers.* `https://sbert.net/docs/package_reference/sentence_transformer/sampler.html` (visited on 06/30/2025).

[17] Weijie Jiang and Zachary A Pardos. "Evaluating Sources of Course Information and Models of Representation on a Variety of Institutional Prediction Tasks." In: *International Educational Data Mining Society* (2020).

[18] Shamrock Solutions LLC. *Transfer Credit Automation: How Universities Are Simplifying Course Equivalency.* Overland Park, KS, USA. `https://www.shamrocksolutionsllc.com/post/transfer-credit-automation-universities` (visited on 06/30/2025).

[19] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG]. https://arxiv.org/abs/1711.05101.

[20] H Ma et al. "Course recommendation based on semantic similarity analysis". In: *2017 3rd IEEE International Conference on Control Science and Systems Engineering*. 2017, pp. 638–641.

[21] Z. A Pardos, H Chau, and H Zhao. "Data-assistive course-to-course articulation using machine translation". In: *Proceedings of the Sixth Conference on Learning@ Scale*. 2019, pp. 1–10.

[22] Zachary Pardos, Hung Chau, and Haocheng Zhao. "Data-Assistive Course-to-Course Articulation Using Machine Translation". In: (July 2019). DOI: 10.1145/3330430.3333622.

[23] Zachary A. Pardos, Hung Chau, and Haocheng Zhao. "Data-Assistive Course-to-Course Articulation Using Machine Translation". In: *Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale*. L@S '19. Chicago, IL, USA: Association for Computing Machinery, 2019. ISBN: 9781450368049. DOI: 10.1145/3330430.3333622. https://doi.org/10.1145/3330430.3333622.

[24] Zachary A. Pardos, Zihao Fan, and Weijie Jiang. *Connectionist Recommendation in the Wild: On the utility and scrutability of neural networks for personalized course guidance*. 2018. arXiv: 1803.09535 [cs.AI]. https://arxiv.org/abs/1803.09535.

[25] Zachary A. Pardos, Zihao Fan, and Weijie Jiang. “Connectionist recommendation in the wild: on the utility and scrutability of neural networks for personalized course guidance”. In: *User Modeling and User-Adapted Interaction* 29.2 (Apr. 2019), pp. 487–525. ISSN: 0924-1868. DOI: 10.1007/s11257-019-09218-7. https://doi.org/10.1007/s11257-019-09218-7.

[26] Stephen Porter. “Assessing Transfer and Native Student Performance at Four-Year Institutions”. In: *39th Annual Forum of the Association for Institutional Research.* June 1999.

[27] Regents of the University of California, The. *Fall enrollment at a glance.* 2024. https://www.universityofcalifornia.edu/about-us/information-center/fall-enrollment-glance (visited on 09/13/2024).

[28] Nils Reimers and Iryna Gurevych. “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing.* Association for Computational Linguistics, Nov. 2019. https://arxiv.org/abs/1908.10084.

[29] Natenaile Asmamaw Shiferaw et al. *BERT-Based Approach for Automating Course Articulation Matrix Construction with Explainable AI.* 2024. arXiv: 2411.14254 [cs.LG]. https://arxiv.org/abs/2411.14254.

[30] Sharon Slade and Paul Prinsloo. “Learning Analytics: Ethical Issues and Dilemmas”. In: *American Behavioral Scientist* 57.10 (2013), pp. 1510–1529. DOI: 10.

1177/0002764213479366. eprint: https://doi.org/10.1177/0002764213479366. https://doi.org/10.1177/0002764213479366.

[31] The National Task Force on the Transfer and Award of Credit. *Reimagining Transfer for Student Success*. Report to Congressional Requesters. American Council on Education, Mar. 2020. https://www.gao.gov/products/gao-17-574.

[32] United States Government Accountability Office. *Higher Education: Students Need More Information to Help Reduce Challenges in Transferring College Credits*. Report to Congressional Requesters GAO-17-574. United States Government Accountability Office, Aug. 14, 2017. https://www.gao.gov/products/gao-17-574.

[33] Yinuo Xu and Zach A. Pardos. "Extracting Course Similarity Signal using Subword Embeddings". In: *Proceedings of the 14th Learning Analytics and Knowledge Conference*. LAK '24. Kyoto, Japan: Association for Computing Machinery, 2024, pp. 857–863. ISBN: 9798400716188. DOI: 10.1145/3636555.3636903. https://doi.org/10.1145/3636555.3636903.

[34] Jian Yu et al. "Deep metric learning with dynamic margin hard sampling loss for face verification". In: *Signal, Image and Video Processing* 14.4 (June 2020), pp. 791–798. ISSN: 1863-1711. DOI: 10.1007/s11760-019-01612-3. https://doi.org/10.1007/s11760-019-01612-3.