Order N^o : ../../..

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY OF KASDI MERBAH OUARGLA

Faculty of New Technologies of Information and Communication
Department of Computer science and technology of Information

THESIS

Thesis submitted in partial fulfillment of the requirements for the degree of

3^{rd} **Cycle LMD Doctorate**

Option : **Computer Vision**

By : **Mebarka ALLAOUI**

Theme

Embedding techniques and their application to Computer vision

Publicly defended on :15/02/2023 before the jury composed of :

Djamel SAMAI	*Pr.*	*at Ouargla University*	*President*
Mohammed Lamine KHERFI	*Pr.*	*at Ouargla University*	*Supervisor*
Abdelhakim CHERIET	*MCA*	*at Ouargla University*	*Co-supervisor*
Belal KHALDI	*MCA*	*at Ouargla University*	*Examiner*
Abdelkader LAOUID	*Pr.*	*at El Oued University*	*Examiner*
Younes GUELLOUMA	*MCA*	*at Laghouat University*	*Examiner*
Abdelhamid BOUCHACHIA	*Pr.*	*at Bournemouth University, UK*	*Invited*
Samir Brahim BELHAOUARI	*Pr.*	*at Hamad Bin Khalifa University*	*Invited*
Oussama AIADI	*MCA*	*at Ouargla University*	*Invited*

Année Universitaire 2023/2024


بسم الله الرحمن الرحيم


وقل رب زدني علما

صدق الله العظيم

I dedicate this work especially to the world dearest persons to me, who inspired me and gave me courage and hope, to my lovely ***Mom*** *and* ***Dad****.*
To my dear brothers ***Adnane****,*
Abdsalame*, and* ***Abdelouahed****.*
To My dearest sisters ***Imen*** *and* ***Batoul****.*
To all my family and my friends.
To all those who were giving me any kind of support.

Acknowledgments

First of all, I thank ***God*** *Almighty for allowing me to reach this scientific level and for giving me the courage and patience to complete my academic achievement.*

I would like to thank my supervisors ***Pr. Muhammad Lamine KHERFI, and Dr. Abdelhakim CHERIET*** *for the help, patience, support, and guidance throughout the period of this research study. My special thanks to my supervisors* ***Pr. Muhammad Lamine KHERFI*** *for his constructive remarks and for instilling in me diligence, sincerity in work, and always looking forward to the best.*

Furthermore, I would like to express my sincere appreciation and gratitude for working with ***Pr. Abdelhamid BOUCHACHIA****. I would like to thank him for the help, guidance, patience, and support that he gave me throughout the hard times.*

In addition, I would like to express my sincere thanks to ***Dr. Oussama AIADI*** *and* ***Pr. Samir Brahim BELHAOUARI****. I thank them for their support and for helping me in the realization of my work.*

I would like to acknowledge all the members of the committee, for accepting and evaluating my thesis work.

Thank you Mom, Dad, all my family and friends. To each and every one of you – Thank you so much.

الملخص

يستخدم تقليل الأبعاد على نطاق واسع في تحليلات التعلم الآلي لأنه يساعد في تحليل مجموعات البيانات الكبيرة عالية الأبعاد وتصورها. على وجه الخصوص ، يمكن أن يساعد بشكل كبير في أداء مهام مثل تجميع البيانات وتصنيفها. في الآونة الأخيرة ، ظهرت طرق التضمين كإتجاه واعد لتحسين دقة التجميع. نتيجة لذلك ، يمكن استخدام تقنيات التضمين والتجميع القوية للكشف عن مشاكل الحياة الواقعية.

تحتوي هذه الرسالة على اربع مساهمات اساسية : (1) لقد بحثنا في كيفية تحسين أداء العديد من خوارزميات التجميع باستخدام واحدة من أنجح تقنيات التضمين. فرضيتنا الرئيسية هي أن تقنية التضمين المستخدمة ستسمح بإيجاد أفضل مشعب تضمين قابل للتجميع. لذلك ، قمنا بتطبيقها كخطوة معالجة مسبقة قبل التجميع ، مما يسمح لخوارزميات التجميع بتحسين أدائها. (2) أجرينا التضمين والتجميع في وقت واحد من خلال صياغة أصلية تسمح بالحفاظ على البنية الأصلية للبيانات في مساحة التضمين وإنتاج مهمة تجميع أفضل. تعتمد خوارزمية التضمين والتجميع المتشعب الموحد (UEC) على وظيفة الخسارة ثنائية الهدف التي تجمع بين تضمين البيانات وتجميعها ، والتي تم تحسينها باستخدام ثلاث طرق مختلف. (3) أثار تفشي وباء كوفيد 19 أدت الى مشكلة جدية. أصبح من الصعب للغاية على الباحثين مواكبة أحدث التطورات العلمية بسبب العدد الكبير من المقالات العلمية المنشورة في فترة قصيرة. قدمنا أداة ذكية تعتمد على التعلم الآلي ، والتي تنظم تلقائيًا مجموعة بيانات كبيرة من المؤلفات العلمية المتعلقة بكوفيد 19 وتصورها بطريقة تساعد هؤلاء الأشخاص على التنقل بسهولة من خلال مجموعة البيانات هذه وتحديد موقع المستندات المطلوبة بسهولة. تتم معالجة المستندات أولاً مسبقًا وتحويلها إلى ميزات رقمية. بعد ذلك ، يتم تمرير هذه الميزات من خلال جهاز تشفير تلقائي عميق لتقليل الضوضاء متبوعًا بتقنية التقريب والإسقاط المتشعب الموحد لتقليل أبعادها إلى مساحة ثنائية الأبعاد. يتم بعد ذلك تجميع البيانات المتوقعة باستخدام خوارزمية التجميع التكتلي. يتبع ذلك خطوة نمذجة الموضوع ، والتي أجريناها باستخدام تخصيص Latent Dirichlet Allocation (LDA) لتعيين تسمية لكل مجموعة. (4) تقترح بنية جديدة للتعلم العميق لتنظيم مجموعة البيانات الكبيرة من المؤلفات العلمية المتعلقة بكوفيد 19 . تكمن الفكرة الأساسية للبنية المقترحة في تدريب المشفر التلقائي باستخدام دالة الهدف ذات شقين تتضمن طرفين مختلفين. الطرف الأول مخصص لتقييم التمثيل الكامن ، بينما يستخدم الطرف الثاني لتحقيق مهام التجميع. بعد ذلك ، نستخدم خوارزمية Latent Dirichlet Allocation (LDA) لتحليل موضوعات المستندات.

الكلمات المفتاحية : **التعلم الآلي ، تقليل الأبعاد ، التضمين ، التجميع ، التعلم المشترك.**

Abstract

Dimensionality reduction is a commonly employed technique in the field of machine learning and analytics, as it aids in the examination and representation of expansive datasets characterized by a multitude of dimensions. This approach is precious for enhancing the effectiveness of tasks such as data clustering and classification. Recently, embedding methods have emerged as a promising direction for improving clustering accuracy. As a result, robust embedding and clustering techniques can be used to resolve real-life problems.

The contribution of the present dissertation is fourfold : (1) We explored ways to enhance the performance of several clustering algorithms by employing one of the most effective embedding techniques available. Our central hypothesis posits that the chosen embedding technique can enable the discovery of the optimal clusterable embedding manifold. Consequently, we utilized it as a preprocessing step prior to clustering, thereby enabling the clustering algorithms to enhance their performance. (2) We performed embedding and clustering simultaneously through an original formulation, which allows for preserving the data's original structure in the embedding space and producing a better clustering assignment. The unified manifold embedding and clustering (UEC) algorithm is based on a bi-objective loss function that combines data embedding and clustering, which is optimized using three different ways : 1) Comma Variant, 2) Plus Variant, and 3) Light Plus Variant. (3) The onset of the COVID-19 pandemic posed a significant challenge, making it increasingly arduous for researchers to keep abreast of the latest scientific advancements, given the rapid influx of scientific articles. To address this issue, we introduced an intelligent tool rooted in Machine Learning. This tool automatically organizes a vast repository of scientific literature pertaining to COVID-19 and presents it in a manner that facilitates easy navigation and swift document retrieval. The initial step involves preprocessing and transforming the documents into numerical features. Subsequently, these features undergo dimensionality reduction into a 2D space through a deep denoising autoencoder followed by the Uniform Manifold Approximation and Projection technique (UMAP). The projected data is then clustered using the Agglomerative Clustering Algorithm. Finally, we employ Latent Dirichlet Allocation (LDA) for topic modeling, assigning a label to each cluster. (4) We propose an innovative deep learning framework designed to structure the extensive collection of scientific literature pertaining to COVID-19. The fundamental concept underlying this architecture revolves around the training of the autoencoder, which utilizes a two-fold objective function comprising distinct terms. The first term is devoted to assessing the latent representation, while the second is used to achieve the clustering assignments. Afterward, the Latent Dirichlet Allocation (LDA) is used as topic modeling techniques.

Keywords : machine learning, dimensionality reduction, embedding, clustering, joint learning.

Résumé

La réduction de la dimensionnalité est largement utilisée dans l'analyse de l'apprentissage automatique, car elle permet d'analyser et de visualiser de grands ensembles de données de grande dimension. En particulier, cela peut considérablement aider à effectuer des tâches telles que le regroupement et la classification des données. Récemment, les méthodes d'intégration sont apparues comme une direction prometteuse pour améliorer la précision du clustering. En conséquence, des techniques d'intégration et de regroupement robustes peuvent être utilisées pour révéler des problèmes réels.

La contribution de la présente thèse est quadruple : (1) nous avons étudié comment améliorer les performances de plusieurs algorithmes de clustering en utilisant l'une des techniques d'intégration les plus performantes. Notre hypothèse principale est que la technique de plongement utilisée permettrait de trouver la meilleure variété de plongement clusterisable. Par conséquent, nous l'avons appliqué comme étape de prétraitement avant le clustering, permettant aux algorithmes de clustering d'améliorer leurs performances. (2) Nous avons effectué l'intégration et le regroupement simultanément grâce à une formulation originale qui permet de préserver la structure originale des données dans l'espace d'intégration et de produire une meilleure affectation de regroupement. L'algorithme d'intégration et de regroupement de variétés unifiées (UEC) est basé sur une fonction de perte bi-objectif qui combine l'intégration et le regroupement de données, qui est optimisée de trois manières différentes : 1) Variante virgule, 2) Variante plus et 3) Variante légère plus . (3) L'apparition de la pandémie de COVID-19 a soulevé un grave problème, qui est devenu extrêmement difficile pour les chercheurs de se tenir au courant des dernières avancées scientifiques en raison d'un grand nombre d'articles scientifiques dans une courte période. Nous avons présenté un outil intelligent basé sur l'apprentissage automatique, qui organise automatiquement un grand ensemble de données de la littérature scientifique liée au COVID-19 et les visualise de manière à aider ces personnes à naviguer facilement dans cet ensemble de données et à localiser facilement les documents recherchés. Les documents sont d'abord pré-traités et transformés en caractéristiques numériques. Ensuite, ces caractéristiques sont passées à travers un autoencodeur de débruitage profond suivi de la technique UMAP (Uniform Manifold Approximation and Projection) pour réduire leur dimensionnalité dans un espace 2D. Les données projetées sont ensuite regroupées avec l'algorithme de regroupement agglomératif. Ceci est suivi d'une étape de modélisation de sujet, que nous avons effectuée en utilisant la Latent Dirichlet Allocation (LDA) pour attribuer une étiquette à chaque cluster. (4) nous proposons une nouvelle architecture d'apprentissage en profondeur pour organiser le vaste ensemble de données de la littérature scientifique liée au COVID-19. L'idée centrale de l'architecture proposée réside dans

l'entraînement de l'auto-encodeur à l'aide d'une fonction objectif double qui incorpore deux termes différents. Le premier terme est consacré à l'évaluation de la représentation latente, tandis que le second est utilisé pour réaliser les affectations de clustering. Ensuite, nous utilisons la Latent Dirichlet Allocation (LDA) pour analyser les sujets du document.

Mots-clés : apprentissage automatique, réduction de la dimensionnalité, incorporation, regroupement, apprentissage conjoint.

Contents

Contents ix

List of Figures xiii

List of Tables xiv

I Context And Motivations xvi

1 General Introduction 1

1.1 Motivations . . . 2

1.1.1 The application of embedding and clustering techniques . . . 3

1.2 Contributions . . . 3

1.3 Organization of the Dissertation . . . 4

2 State of the art 6

2.1 Introduction . . . 6

2.2 Embedding and dimensionality reduction techniques . . . 6

2.3 Clustering techniques . . . 7

2.3.1 Partitioning clustering . . . 7

2.3.2 Distribution clustering . . . 8

2.3.3 Hierarchical clustering . . . 8

2.3.4 Density-based clustering . . . 9

2.3.5 Manifold clustering . . . 9

2.3.6 Deep manifold clustering . . . 9

2.3.7 Deep clustering . . . 10

2.4 The application of embedding and clustering in COVID-19 problems . . . 12

2.5 Requirements . . . 13

2.5.1 Datasets . . . 13

2.5.2 Evaluation Metrics . . . 15

2.6 Conclusion . . . 17

II Contributions **18**

3 Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study **19**
3.1 Introduction . . . 19
3.2 UMAP embedding technique for dimensionality reduction . . . 20
3.3 Proposed Method . . . 22
3.4 Experiments . . . 22
3.4.1 Requirements . . . 23
3.4.2 Results . . . 23
3.5 Conclusion . . . 26

4 Unified Embedding and Clustering **27**
4.1 Introduction . . . 27
4.2 Unification of Manifold Embedding and Clustering . . . 28
4.3 Formulation of the algorithm ingredients . . . 30
4.3.1 Computation of the affinity matrix P . . . 30
4.3.2 Initialization . . . 31
4.3.3 Computation of the affinity Matrix Q . . . 31
4.3.4 Computation of the soft assignments S . . . 32
4.3.5 Computation of the auxiliary target distribution T . . . 32
4.3.6 Formulation of the optimization problem . . . 33
4.4 Optimization of the objective function . . . 33
4.4.1 Comma (Sequential) variant . . . 34
4.4.2 Plus (Combined) variant . . . 35
4.4.3 Light Plus variant . . . 36
4.4.4 Implementation . . . 37
4.5 Experiments and discussion . . . 37
4.5.1 Requirements . . . 38
4.5.2 **Comparison of the variants** . . . 38
4.5.3 **Effect of the number of clusters** . . . 40
4.5.4 **Embedding space Initialization** . . . 41
4.5.5 **Preserving the local and the global structures of the data** . . . 43
4.5.6 **Effect of Alpha and Beta** . . . 45
4.5.7 Comparative Study . . . 47
4.5.8 Evaluation on Challenging datasets . . . 49
4.5.9 Qualitative results . . . 50
4.6 Conclusion . . . 53

5 A Machine Learning-Based Tool for Exploring COVID-19 Scientific Literature **54**

5.1 Introduction . . . 54
5.2 Proposed Method . . . 55
5.2.1 Pre-processing . . . 57
5.2.2 Feature Extraction . . . 57
5.2.3 Dimensionality reduction . . . 57
5.2.4 Projecting and Clustering . . . 58
5.2.5 Topic Modeling . . . 59
5.2.6 Visualization Tool . . . 59
5.3 EXPERIMENTAL EVALUATION . . . 60
5.3.1 Evaluation Protocol . . . 60
5.3.2 First experiment: Homogeneity of our clusters . . . 60
5.3.3 Second experiment: Internal Clustering performance of our algorithm 62
5.3.4 ELBOW method to find the correct number of clusters . . . 62
5.4 Conclusion . . . 63

6 A novel two-fold loss function for data clustering and reconstruction: application to document analysis 64
6.1 Introduction . . . 64
6.2 Proposed Method . . . 65
6.2.1 Dataset pre-processing . . . 66
6.2.2 Dimensionality Reduction and Cluster assginments . . . 67
6.2.3 Topic Modeling . . . 72
6.3 Experimental results . . . 73
6.3.1 Experimental setup . . . 73
6.3.2 Comparative Study . . . 74
6.3.3 Case study using CORD-19 dataset . . . 74
6.4 Conclusion . . . 83

7 General Conclusion and Perspectives 86
7.1 General Conclusion . . . 86
7.2 Perspectives . . . 87

III Appendices 90

A Appendix A: The derivation of the embedding objective function (Cross-Entropy) 91

B Plus variant derivations 94
B.1 Derivative of the total loss function w.r.t. datapoints . . . 98
B.2 The derivative of Objective Function w.r.t. the centroids . . . 102

C Light plus variant derivations **107**
C.1 Derivation of the KL divergence w.r.t. datapoints 107
C.2 Derivative of the total loss function w.r.t. datapoints 108
C.3 The derivative of Objective Function w.r.t. the centroids 108

D Definition of the values domain of the CE and KL divergence gradients w.r.t. datapoints **111**
D.1 Defining the interval values of the CE gradient w.r.t. datapoints 111
D.2 Defining the values domain of the KL divergence gradient w.r.t. datapoints 112
D.2.1 The derivative of the KL divergence Plus variant 112
D.2.2 The derivative of the KL divergence Light Plus variant 112

Bibliography **113**

List of Figures

2.1 Categorization of Clustering Techniques. 8

3.1 Application of clustering algorithms on the embedding space of UMAP. . . 22
3.2 Visualization of K-Means applied to all five datasets. 23

4.1 The effect of changing the number of clusters measured by Silhouette score. 41
4.2 The effect of changing the number of clusters measured by Accuracy and NMI. 42
4.3 Preservation of the local structure using k-NN graph. 44
4.4 Preservation of the global structure using CE function. 45
4.5 visualization of challenging datasets: Original vs. The UEC visualization. . 51
4.6 The visualization of USPS and MNIST datasets using the UEC against ISOMAP and t-SNE. 52

5.1 The steps of exploring the documents related to COVID-19. 56
5.2 Visualization of the literature related to COVID-19. 59
5.3 Result of clustering algorithms in terms of accuracy. 61
5.4 ELBOW method to find the best value of K: the number of clusters. . . . 63

6.1 DJRC's architecture. 66
6.2 Topic Modeling step using LDA. 67
6.3 Latent Dirichlet Allocation description. 73
6.4 Count of words in papers. 76
6.5 Evaluating the performance of the proposed algorithm in terms of DB and SC when varying the number of clusters. 77
6.6 The error of the clustering and reconstruction loss per epochs. 78
6.7 Most important features using Unsupervised to Supervised method for the clusters 0, 1, 2, and 3. 79
6.8 Most important features using Unsupervised to Supervised method for the clusters 4, 5, 6, and 7. 80
6.9 The number of documents shared the same topics. 81
6.10 Measuring the performance of LDA using C_v score while varying the number of topics. 82
6.11 Visualization of CORD-19 dataset using UMAP. 85

List of Tables

2.1 Dataset statistics. . . . 14

3.1 Comparison between the different clustering algorithms on the five datasets according to the accuracy measure. . . . 24
3.2 Comparison between the different clustering algorithms on the five datasets according to the NMI measure . . . 25
3.3 The execution time before and after applying UMAP on the different clustering algorithms on the five datasets. . . . 25

4.1 Description of all used symbols . . . 28
4.2 Evaluating the performance of the UEC's three variants in Accuracy (ACC) and NMI. . . . 40
4.3 Evaluating the performance of the UEC's three variants in terms of Execution time (in seconds). . . . 40
4.4 Clustering quality using the Davies-Bouldin index. . . . 41
4.5 Clustering quality using the Silhouette index. . . . 41
4.6 Clustering quality using the Calinski-Harabaz index. . . . 43
4.7 Effect of center initialization on the performance. . . . 43
4.8 Effect of the embedding space initialization on the performance. . . . 43
4.9 Effect of α and β on the performance of the algorithm measured by accuracy. 46
4.10 Effect of α and β on the performance of the algorithm measured by NMI. . 46
4.11 Effect of different values of α and β on the algorithm's performance measured by the accuracy. . . . 47
4.12 Effect of different values of α and β on the performance of the algorithm measured by the NMI. . . . 47
4.13 UEC vs. other baselines: accuracy scores. . . . 48
4.14 UEC vs. other baselines: NMI scores. . . . 49
4.15 Evaluating the performance of our three variants against the other techniques in terms of Execution time (in seconds). . . . 49
4.16 Evaluating the performance of UEC's variants against the other techniques in terms of Accuracy (ACC) and NMI. . . . 50

5.1 Comparison with different algorithms in terms of Davies-Bouldin (DB), Calinski-Harabasz (CH), and Silhouette Coefficient (SC). . . . 62

6.1 DJRC vs. other baselines: accuracy scores. 75
6.2 DJRC vs. other baselines: NMI scores. 75
6.3 The number of papers in each language. 75
6.4 Comparison with different algorithms in terms of Davies-Bouldin (DB) and Silhouette Coefficient (SC). 77
6.5 The descriptive terms of each topic per cluster. 84

Part I

Context And Motivations

Chapter 1

General Introduction

Machine learning (ML) is a growing field in which machines are trained on data to learn specific tasks automatically. In ML, several sub-fields, such as classification, clustering, image retrieval, or object recognition, take features extracted from data as inputs. Those features could be hand-crafted or automatically learned, e.g., Dimensionality reduction algorithms.

Dimensionality reduction (DR) can be defined as transforming the dimensions of a larger dataset into fewer dimensions while retaining the essential features. In machine learning, dimensionality reduction techniques are employed to improve predictive model accuracy, although their effectiveness may vary [1]. These techniques aim to fulfill a range of crucial criteria, including the capacity to reveal the intricate structure of high-dimensional data, maintain proximity relationships, scale efficiently in terms of computation, withstand data noise and outliers, and offer practical usability. The key questions are how to locate this low-dimensional structure, how to use it, and what presumptions are made when extracting it from data. As a result, several methods are proposed to answer these problems [2].

In the context of machine learning and dimensionality reduction and expansion, embedding refers to a numerical representation of objects in a vector space. These vector representations are designed in a way that captures semantic relationships and similarities between the objects [3].

Neural embedding (NE) is a sub-field of embedding that learns data representation

by neural networks while modeling or solving a given problem. Neural networks (NN) have been successfully applied for text modeling (word embedding) and extending to more complex domains, e.g., graph embedding [4].

Our study focuses mainly on using dimensionality reduction and embedding techniques to solve problems related to data clustering. Clustering is a fundamental operation in unsupervised machine learning that has received considerable attention from various research communities.

1.1 Motivations

Clustering algorithms struggle with high-dimensional datasets due to the curse of dimensionality, making them suffer many challenges. These challenges encompass various issues, such as the convergence of point distances, the hindered visualizability of outputs, the distortion of point locations due to correlation, data sparsity, and the problem of local feature relevance. Thus, many clustering algorithms will fail to converge in these situations and have significant computational complexity when dealing with massive datasets [5]. For instance, the K-means algorithm [6] exhibits a time complexity of $O(N^2)$, rendering it impractical for application as the number of rows (N) increases.

In addition, Several clustering algorithms adopt an approach of initializing computation with random centroids. This approach introduces variability, resulting in different solutions based on the choice of cluster numbers and the initial placement of centroids. The presence of numerous local solutions compromises the reproducibility of the analysis. An optimal solution may not even exist in certain instances, and the clustering algorithm may yield multiple local solutions contingent upon the initial centroid placements.

The mentioned challenges are addressed by operating Clustering algorithms on the projection space of the dataset using one of the embedding techniques. The choice of features greatly influences the performance of the clustering algorithm. Thus, applying adequate embedding techniques on the raw data allows for finding a good low-dimensional space, which could help clustering algorithms do their job.

Recent studies have tended to develop algorithms that learn dimension reduction and clustering simultaneously. Research has demonstrated that bi-objective learning algo-

rithms outperform approaches that sequentially apply dimension reduction and clustering methods [7, 8, 9, 10].

1.1.1 The application of embedding and clustering techniques

The embedding and clustering techniques could solve real-life problems, such as the challenges of the COVID-19 epidemic.

The emergence of the COVID-19 pandemic has triggered a swift proliferation of scientific articles directly or indirectly related to COVID-19. While this surge in literature is undeniably beneficial, it is not without its challenges. In fact, it has become increasingly difficult, if not unfeasible, for researchers or decision-makers to remain current with the latest scientific developments in this domain, sift through the vast number of recently published articles, or locate specific articles that address particular questions or aspects of COVID-19.

Conventional document organization and search tools are ill-suited for this purpose, as they were not originally designed to handle this type of content [11]. Consequently, they may lack precision and fail to fulfill the particular requirements of scientists and decision-makers who are seeking COVID-19 literature. Therefore, there is a pressing and immediate demand for the development of specialized methods tailored to this specific body of literature.

1.2 Contributions

The primary contributions of this thesis can be succinctly summarized as follows:

1. Motivated by the fact that the performance of the Clustering algorithms can be improved when they are applied to features extracted using embedding techniques. And this is what we do precisely in our first contribution. We applied Several well-known clustering algorithms on low-dimensional space produced by a recent manifold embedding technique, Uniform Manifold Approximation and Projection (UMAP).
2. The key contributions of this thesis encompass the introduction of a pioneering manifold-based learning algorithm. This algorithm adeptly accomplishes the dual

task of learning the embedding space and clustering the data simultaneously. This joint optimization approach, addressing both the embedding and clustering objectives, results in enhanced preservation of the original data's similarities and improved clusterability within the embedding space. The theoretical underpinnings of the optimization procedure are intricately linked with the proposed UEC variants.

3. One of the applications of embedding and clustering techniques is to organize and visualize COVID-19-related documents. We introduced a tool that harnesses cutting-edge embedding techniques, specifically UMAP [12], and a deep autoencoder for dimensionality reduction [13]. The choice of the agglomerative clustering algorithm was made due to its commendable performance. To address the issue of clustering algorithms organizing similar articles into clusters without providing labels, we incorporated a crucial topic modeling step. In this step, we employed Latent Dirichlet Allocation (LDA) to identify the most frequently occurring keywords within each cluster. These keywords were then utilized to assign labels to groups of documents, thereby facilitating user comprehension of the content within each cluster.

4. We introduce an innovative deep learning-based framework designed for the exploration and organization of a substantial dataset comprising COVID-19-related documents. Our approach seamlessly integrates dimensionality reduction and clustering through a dual-objective function. Specifically, our architecture features a denoising autoencoder with three components: two distinct encoders and one decoder. These three components undergo training using the dual-objective function, where the initial term within the function addresses the Reconstruction loss, with the objective of reducing the dimensionality of the input features. The second term in the function is dedicated to predicting cluster assignments, and it involves the iterative updating of two separate matrices associated with the clean and noisy encoders during training. Subsequently, we employ Latent Dirichlet Allocation (LDA) for topic modeling.

1.3 Organization of the Dissertation

The chapters of this thesis are organized as follows.

Chapter 2: the state of the art of dimensionality reduction and embedding techniques is illustrated in this chapter. Then, the baselines techniques of clustering are detailed in this chapter. In addition, some proposed approaches to solve the problem of the enormous number of COVID-19-related documents using embedding and clustering techniques.

Chapter 3: Firstly, we present our method for improving the performance of clustering algorithms using UMAP. Then, we validate our method in the second section using the evaluation metrics.

Chapter 4: Our proposed techniques, Unified Embedding and Clustering (UEC) is presented in this chapter. Then, the experimental results of UEC against the state of art methods are demonstrated to evaluate the robustness and efficiency of the proposed method.

Chapter 5: Chapter 6 demonstrates our solution to exploring and organizing COVID-19-related documents. Then, the experimental part presents and discusses the results obtained from our method against some other previous methods.

Chapter 6: In the same context of solving the problem of organizing COVID-19-related documents, we present our novel method. The first section describes the components of our model. Then, the results of our novel method against the state of art methods are demonstrated and discussed in the experimental part.

Chapter 7: We conclude our thesis and provide perspectives that we will consider, in chapter 7.

Chapter 2

State of the art

2.1 Introduction

In this thesis, we focused mainly on two sub-fields of machine learning: dimensionality reduction and clustering. In addition, as mentioned in the Introduction chapter, we mention some embedding and clustering techniques used to solve the challenges posed by the COVID-19 epidemic.

This chapter presents an overview of the relevant state-of-the-art for each sub-field (i.e., dimensionality reduction, clustering, and their application to COVID-19 challenges). We categorized the previous studies of each sub-field according to their way of work. Furthermore, we describe in this chapter the used data sets and evaluation metrics in the different contributions.

In following, Section 2.2 is an overview of Embedding and dimensionality reduction techniques. In section 2.3, we introduce the clustering techniques. Section 2.4 discusses the application of Embedding and clustering techniques to COVID-19 challenges. The datasets and evaluation metrics used in our studies are described in section 2.5. Section 2.6 is the conclusion of the chapter.

2.2 Embedding and dimensionality reduction techniques

Dimensionality reduction (DR) techniques are used to project the data into a lower dimensional space. According to [14], DR methods can be categorized into projection, manifold embedding, and deep learning methods.

Projection techniques like Principal Component Analysis (PCA) [15] aim to achieve a linear data transformation into a new feature space. Nonetheless, their linearity can lead to sub-optimal performance when dealing with datasets characterized by non-linear relationships.

Manifold Embedding (ME) methods provide alternatives that can address the challenges related to the non-linearity of data, with a focus on local or global structures. Examples of ME techniques include Isomap [16], along with its precursor Multidimensional Scaling (MDS), Locally Linear Embedding (LLE) [17], and the more recent UMAP algorithm [12]. These methods leverage the distances between original data points to enhance their understanding of the underlying structure and preserve similarity among data points in lower-dimensional spaces. Their goal is to capture the most relevant information, but it's important to note that they may suffer from discriminative information loss, which can impact clustering performance.

Deep dimensionality reduction (DR) methods have introduced a range of autoencoder architectures [18, 19] and convolutional neural network (CNN) architectures [20, 21]. These advancements have demonstrated substantial enhancements in numerous computer vision tasks [22, 23, 24, 25, 26]. While deep DR methods excel at handling large-scale datasets, it's important to note that they require substantial computational resources and ample memory capacity.

2.3 Clustering techniques

The field of clustering has undergone extensive examination in the literature, yielding numerous algorithms. Those clustering algorithms can be categorized into seven families depending on their conceptual idea [27]: 1) Partitioning, 2) Distribution, 3) Hierarchical, 4) Density-based, 5) Manifold clustering, 6) Deep manifold clustering, 7) and Deep clustering. See figure 2.1.

2.3.1 Partitioning clustering

Among the various types of clustering algorithms, partitioning clustering algorithms are the most widely employed. One of the renowned methods within this category is

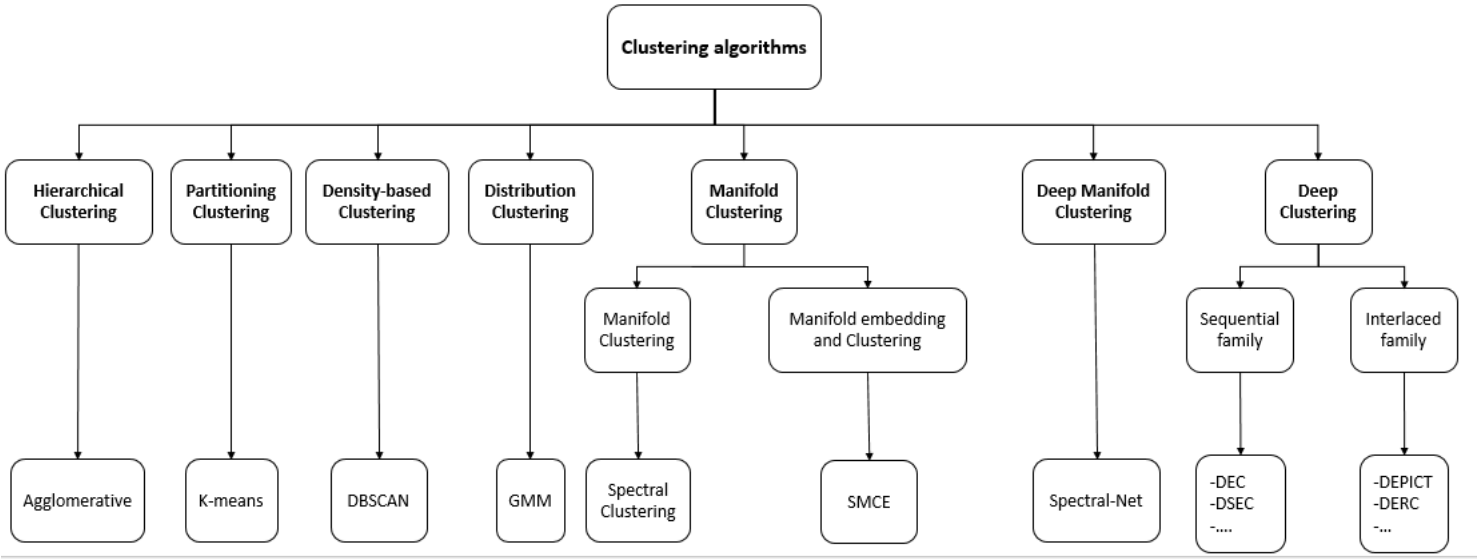

Figure 2.1 – Categorization of Clustering Techniques.

K-means [6, 28], which divides data points into multiple clusters using centroids. Each data point is allocated to a cluster according to its squared distance from the centroid. Despite the speed and efficiency of these algorithms, they are susceptible to the initial parameter settings.

2.3.2 Distribution clustering

Distribution-based clustering methods operate on the premise that all data points are potential members of all clusters based on a probability assignment. With a central point as a reference, the likelihood of belonging to that cluster diminishes as a data point's distance from that center grows. Within this category, the Gaussian Mixture Model (GMM) [29] is an example.

2.3.3 Hierarchical clustering

Within the realm of Hierarchical clustering, Agglomerative clustering [30] stands out as a method that does not necessitate the pre-specification of the number of clusters. This algorithm begins by treating each data point as an independent cluster. Subsequently, it combines similar clusters in each iteration until no further merging can occur. The Agglomerative clustering algorithm can be summarized as follows:

1. Computation of the proximity matrix.
2. Each data point is Considered as a cluster.

3. Repeat:
    — Merge the two closest clusters and update the proximity matrix Until only a single cluster remains, [30].

The main drawback of Hierarchical clustering algorithms is that they encounter tremendous computational complexity when working with massive datasets.

2.3.4 Density-based clustering

Density-based clustering algorithms, like DBSCAN [31, 32], operate on the fundamental concept that data is grouped based on regions characterized by dense concentrations of data points, enveloped by regions of lower densities. A notable advantage of these algorithms is their ability to accommodate clusters of various shapes. However, a significant drawback is their failure to assign outliers to clusters, resulting in their exclusion from the clustering process. Moreover, density-based clustering algorithms exhibit inefficiency and high computational complexity when handling large-scale datasets in all cases.

2.3.5 Manifold clustering

Manifold clustering (MC) integrates discriminative dimensionality reduction with clustering techniques [33, 34]. MC methods involve the computation of a similarity matrix to project nonlinear data onto low-dimensional manifolds. Spectral clustering, a prominent manifold approach, represents data using a graph wherein the affinity matrix captures similarities between data points. This affinity matrix is then leveraged to identify clusters based on the connectedness of graph components. Spectral clustering can be likened to a kernelized version of K-Means [35]. Another manifold-based approach, Sparse Manifold Clustering and Embedding (SMCE) [36], establishes connections between each data point and its nearby neighbors, assigning appropriate weights. Nevertheless, while this approach enhances embedding, it does not necessarily improve clustering performance [37].

2.3.6 Deep manifold clustering

In the realm of deep manifold clustering, as exemplified by Spectral-Net [38], the objective is to train a network that transforms the input data into the eigenspace of the

graph Laplacian matrix. Following this transformation, a Siamese network is employed to learn the weights of connections between nodes in the graph, culminating in the use of k-means for the final clustering step. Nevertheless, manifold clustering techniques grapple with substantial computational demands when handling extensive datasets. These methods also possess limited representational capabilities, often resulting in the underfitting of the semantic aspects of natural data's similarity matrix [39].

2.3.7 Deep clustering

The last family is Deep clustering. In recent clustering research, there has been a growing exploration of deep learning techniques that combine dimensionality reduction (DR) and clustering, commonly referred to as "deep clustering" [40, 41]. We divide the deep clustering methods into two classes according to how they optimize both loss functions: sequential and interlaced deep algorithms.

The sequential deep methods first perform the optimization of the embedding loss and then that of the clustering one. Deep Embedding and Clustering (DEC) belongs to this class [7]. In DEC, the autoencoder is pre-trained and optimized using the Kullback-Leibler divergence function by imposing some constraints to optimize the clustering assignments. The self-learning strategy of DEC could mislead the learning process by producing non-representative features of data clustering [39]. Deep Semantic Embedding and Clustering (DSEC) [42] works similarly to DEC. DSEC concatenates the training data's original feature and semantic space after the pre-training step.

The main drawback of the sequential deep clustering methods is that discarding the decoder part makes less discrimination in the latent space. To mitigate the impact of the random discriminative features during the clustering phase, the Autoencoder-based clustering techniques retain the reconstruction loss.

Within the class of interlaced deep algorithms, the optimization of both embedding and clustering objective functions occurs in an alternating fashion until a convergence criterion is satisfied.

DEeP Embedded RegularIsed ClusTering (DEPICT)[8] comprises a convolutional autoencoder and a single clustering layer, which jointly learn latent features and the distri-

bution of cluster assignments. DEPICT is optimized through the minimization of both the reconstruction error and the relative entropy between the distributions of cluster assignments and their priors. On the other hand, Deep Embedded Dimensionality Reduction Clustering (DERC)[10] combines a deep autoencoder with the t-SNE embedding method [43] to acquire data representations. Cluster centers are initialized using a Gaussian Mixture Model (GMM), and a probability-based triplet loss measure is employed to fine-tune the model and enhance its clustering performance.

In the Joint Unsupervised LEarning (JULE) approach [44], a CNN is trained using an agglomerative clustering loss. During each iteration, hierarchical clustering is performed in the forward pass using an affinity measure, and the data representation is optimized in the backward pass of the CNN. However, JULE faces limitations related to computational and memory complexities associated with computing the affinity matrix generated by the agglomerative clustering loss function.

On the other hand, the Information Maximizing Self-Augmented Training (IMSAT) technique [45] is centered around data augmentation. In IMSAT, the network is trained to maximize the mutual information between the data and the predicted clusters. The network is also subject to regularization to ensure that the original data's cluster assignment aligns with the augmented data assignment.

Deep Adaptive Image Clustering (DAC) [9] generates the label of the points using a CNN. Then, cosine similarities are calculated between points based on the label of points. Those points are sampled as pairs of points according to the cosine similarities. The CNN is trained using the chosen samples based on the developed binary pairwise classification model.

Combining clustering and reconstruction in the interlaced model loss can cause a trade-off between eliminating the non-discriminative details and preserving all information. We tackled this constraint by introducing an enhanced multi-objective function that simultaneously governs the embedding and clustering processes within a single, unified step.

2.4 The application of embedding and clustering in COVID-19 problems

This research primarily focuses on the automated organization of documents related to COVID-19 literature. Notably, only a limited number of literature studies delve into this intriguing subject matter. However, existing studies can be categorized into two groups based on their employed techniques: handcrafted approaches and deep-based methods.

The first category, which falls under the "handcrafted" methods, encompasses approaches that utilize conventional machine learning techniques such as the Expectation Maximization (EM) algorithm and Principal Component Analysis (PCA). For instance, in a study like [46], diverse focuses within COVID-19-related abstracts were analyzed using a clustering approach. In this approach, Singular Value Decomposition (SVD) was employed for dimensionality reduction, while the Expectation-Maximization algorithm (EM) was used for clustering. However, it's important to note that this study exclusively focused on analyzing abstracts and did not consider the entire document, potentially impacting the accuracy of identifying topics discussed within these documents.

In the paper [47], the authors have introduced COVID-19 LC, a tool designed for organizing and visualizing COVID-19 documents. Initially, the documents undergo preprocessing and are transformed into vectors using the TF-IDF algorithm. Subsequently, Principal Component Analysis (PCA) is employed to perform dimensionality reduction. This step serves as a preparatory stage for the subsequent process, where t-distributed Stochastic Neighbor Embedding (t-SNE) is utilized to represent the documents visually. Following this, an unsupervised K-means algorithm clusters the documents by grouping similar data instances into the same cluster and segregating them from dissimilar ones in other clusters. Finally, Latent Dirichlet Allocation (LDA) is implemented to assign labels to these clusters.

However, it's important to note that relying solely on K-means for clustering can give rise to several challenges. For instance, K-means is known to be ill-suited for datasets with unbalanced clusters, as is often the case with datasets like CORD-19. Moreover, K-means is sensitive to the initial clustering solution and may struggle to effectively handle

outliers and noisy data.

In their work presented in [48], the authors developed a platform for extracting information related to COVID-19 clinical risk variables and subsequently presented the results in clustered form to facilitate knowledge discovery. Their study involved a comparative analysis of two clustering algorithms: spectral clustering and Agglomerative clustering. However, there remains potential for enhancement, particularly in terms of reducing the dimensionality of the document-extracted vectors. Additionally, there is room for improving the clustering component, which could lead to significant enhancements in the outcomes of the proposed model, as suggested in previous studies [49, 50].

Regarding the second category, which comprises deep learning-based methods, there is a notable scarcity of studies employing deep learning for the analysis of COVID-19-related documents. In a study described in [51], the authors proposed an approach for organizing and visualizing documents related to COVID-19 sourced from the CORD-19 dataset. Following a preprocessing step, the dataset's dimensions were reduced using a deep-stacked autoencoder. Subsequently, the reduced dataset was mapped into a 2D space using the Uniform Manifold Approximation and Projection (UMAP) technique [12], with the agglomerative clustering algorithm applied for data clustering. Finally, a topic modeling step was conducted using Latent Dirichlet Allocation (LDA).

2.5 Requirements

In this section, we present the used datasets and the evaluation metrics in our different contributions.

2.5.1 Datasets

In this section, we present 10 datasets that are used in the evaluation of our proposed techniques. Table 2.1 summarizes these datasets.

1. **IRIS**: is a numerical dataset of 150 instances with four dimensions categorized into three different classes.
2. **Spiral**: consists of 300 points with two coordination, which forms three spirals.
3. **Atom** In a 3-dimensional space, Atom is comprised of two clusters. The initial

Table 2.1 – Dataset statistics.

Dataset	Number of samples	Number classes	Feature Vector length
IRIS	150	3	4
Spiral	300	3	2
Atom	400	2	3
EngyTime	4096	2	2
Pen Digits	1797	10	64
USPS	9298	10	256
MNIST	70000	10	784
F-MNIST	70000	10	784
UMIST Face	575	20	10304
CIFAR10	60000	10	1024
Reuters	11228	4	2000
CORD-19	570000		2000

cluster contains a dense core consisting of 400 points, which is surrounded by another distinct set of 400 points forming the second cluster [52].

4. **EngyTime**EngyTime is a dataset featuring 4096 points characterized by two variables: "Engy" and "Time." It forms a two-dimensional mixture of Gaussians and can be partitioned into two distinct clusters [53].
5. **Pen Digits**: Pen Digits dataset is an 8x8 gray image that consists of 1797 samples belonging to 10 classes [54].
6. **USPS**: USPS comprises 9298 images distributed across 10 different classes, with each image being a 16x16 grayscale representation [54].
7. **MNIST**: MNIST encompasses a collection of 70,000 images, each of which represents one of the 10 handwritten digits. These images are all in grayscale and have dimensions of 28x28 pixels [55].
8. **F-MNIST**: The Fashion-MNIST (F-MNIST) dataset comprises 70,000 images, each depicting one of the 10 different fashion products. These images are all grayscale and possess dimensions of 28x28 pixels.
9. **UMIST Face**: is a dataset of faces, which consists of 564 images of 20 individuals.
10. **CIFAR10**: CIFAR10 is a dataset comprising 60,000 samples that are categorized into ten distinct classes. Each sample is represented as a 32x32 RGB image [56].
11. **Reuters**: The Reuters dataset, an English news dataset [57], encompasses a total of

11,228 documents. In alignment with the approach outlined in [7], we have employed four (4) categories or classes while eliminating all documents labeled under multiple root categories. As part of our preprocessing steps, we have removed stop words and leveraged the tf-idf representation of the top 2000 most frequently occurring words.

12. **The COVID-19 Open Research Dataset Challenge (CORD-19)** [58] is a freely accessible dataset comprising more than 570,000 scientific papers related to COVID-19, the SARS-CoV2 virus responsible for it, and other corona-viruses of relevance. Among these papers, over 150,000 come complete with their full text. This extensive dataset was collaboratively compiled by the White House and several research teams, with the overarching aim of encouraging researchers worldwide to employ innovative Machine Learning (ML) techniques, including Natural Language Processing (NLP), to gain deeper insights into these viruses and contribute to efforts in mitigating the pandemic's spread.

2.5.2 Evaluation Metrics

This section presents all the evaluation metrics used in our studies in both domains of clustering algorithms topic modeling.

2.5.2.1 Clustering validation metrics

External validation metrics

1. **Accuracy** We assess the performance of all clustering methods using the Clustering ACCuracy (ACC), which is determined by finding the best alignment between the ground truth labels and the predicted labels.:

$$ACC = max_m \frac{\sum_{i=1}^{n} 1\{GT_i = m(PL_i)\}}{n}$$

where m is one-to-one mapping function between the ground truth GT_i and predicted label PL_i of samples z_i.The Hungarian algorithm can effectively calculate the optimal mapping [59].

2. **Normalized Mutual Information (NMI)** The Normalized Mutual Information (NMI) serves as a normalization of the mutual information score, scaling the results within a range from 0, indicating no mutual information, to 1, signifying a perfect correlation. The formula for NMI is as follows:

$$NMI = \frac{2I(GT_i, PL_i)}{[H(GT_i) + H(PL_i)]}$$

Where $I(GT_i, PL_i)$ is the Mutual Information between GT_i, the ground truth, and PL_i, the predicted label. H(.) is the Entropy.

Internal validation measures We employ three widely recognized internal validation indices, namely Calinski-Harabaz (CH) [60], Silhouette coefficient (S) [61], and Davis-Bouldin (DB) [62].

1. **Davies-Bouldin (DB)** The Davies-Bouldin (DB) index [63] takes into account both the intra-cluster dispersion and the inter-structure of clusters. The DB index produces values within the $[0, +\infty]$ range, where lower values indicate better clustering quality.
2. **Calinski-Harabaz (CK)** The Calinski-Harabaz (CH) index [64] quantifies the separation between clusters. CH index values fall within the $[0, +\infty]$ range, with higher values signifying superior clustering quality.
3. **Silhouette Coefficient (SC)** The Silhouette Coefficient (SC) [65] gauges the quality of clustering. The S index yields values within the range of $[-1, 1]$, where values closer to 1 suggest better clustering outcomes, while values nearer to -1 indicate less desirable results.

2.5.2.2 Topic modeling validation metrics

To assess the effectiveness of topic modeling techniques, we employ the following metric:

— **Coherence Score** C_v: The Coherence Score (C_v) is a metric that quantifies how frequently two words, denoted as w_i and w_j, co-occur within the corpus. It is defined

as follows:

$$C_{UMass}(w_i, w_j) = \log \frac{D(w_i, w_j) + 1}{D(w_i)},$$

where $D(w_i, w_j)$ indicates how many times words w_i and w_j appear together in documents, and $D(w_i)$ is how many time word w_i appeared alone. The greater the number, the better is coherence score.

2.6 Conclusion

We focused our interests in this thesis on two fields, which are embedding and clustering, in addition to their applications to solve other problems. This chapter was divided into four parts. The first part gave an overview of the related state-of-the-art works for dimensionality reduction and embedding. Whereas the second part summarized the existing works for clustering. The third part discussed the different proposed methods to organize COVID-19-related documents. The last part represents the datasets and evaluation metrics used in our contributions. The next chapter presents our first contribution, which is to investigate the possibility of improving the performance of the clustering algorithms using embedding techniques.

Part II

Contributions

Chapter 3

Considerably improving clustering algorithms using UMAP dimensionality reduction technique: a comparative study

3.1 Introduction

This study aims to improve the performance of some well-known clustering algorithms such as k-means [6], Gaussian Mixture Models (GMMs) [29], HDBSCAN [32], and hierarchical algorithms [30]. However, those clustering algorithms are time-consuming and suffering while dealing with large datasets, which require hand-crafted or automatically learned features for datasets [5].

In this chapter, we formulate the following hypothesis: if we apply an adequate embedding on our raw data, i.e., an embedding that allows finding a reasonable distance preserving manifold, then this could help clustering algorithms do their job. One key question was: which embedding technique to apply to find the best embedding manifold?

Several non-linear manifold learning methods exist and can be categorized by their fo-

cus on finding local structures such as T-SNE [43] or global structures such as Isomap [16]. A more recent manifold learning technique is UMAP [12]. UMAP performed better in preserving both the local and global structure than its concurrents [12], and it has proven to meet our needs exactly [66, 67]. In this contribution, Our primary focus was measuring the improvement achieved by each clustering algorithm thanks to applying the UMAP embedding manifold. To validate our method, we conduct several experiments on five datasets. Conducted experiments validate our claim about optimizing clustering algorithms using UMAP.

The rest of this chapter is organized as follows. Section 3.2 is an overview of UMAP. In section 3.3, we introduce our idea. Section 3.4 discusses the experimental results in five image datasets. Section 3.5 concludes the chapter.

3.2 UMAP embedding technique for dimensionality reduction

Uniform Manifold Approximation and Projection (UMAP) is a recently proposed manifold learning method that seeks to represent local structures accurately and better incorporate global structures [12]. Compared to t-SNE, it has several advantages. UMAP has been shown to scale well with large datasets, while t-SNE typically struggles with them. UMAP relies on three hypotheses, namely that 1) the data is uniformly distributed on a Riemannian manifold, 2) the Riemannian metric is locally constant 3) the manifold is locally connected. These assumptions make it possible to represent the manifold with a fuzzy topological structure of high-dimensional data points. The embedding manifold is found by searching for a fuzzy topological structure of low-dimensional data projection. To construct the fuzzy topological structure, UMAP represents the data points by a high-dimensional graph. The constructed high-dimensional graph is a weighted graph, with edge weights representing the likelihood that two points are connected.UMAP uses exponential probability distribution to compute the similarity between high dimensional data points:

$$p_{i|j} = \exp(-\frac{d(x_i, x_j) - \rho_i}{\sigma_i}) \tag{3.1}$$

Where d(x_i,x_j) is the distance between the $i-th$ and $j-th$ data points, and ρ is the distance between $i-th$ data points and its first nearest neighbor. In cases where the weight of the graph between i and j nodes is not equal to the weight between j and i nodes. UMAP uses a symmetrization of the high-dimensional probability:

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i} \tag{3.2}$$

As we said above, the constructed graph is a likelihood graph, and UMAP needs to specify k the number of nearest neighbors:

$$k = 2^{\sum_i p_{ij}} \tag{3.3}$$

Once the high-dimensional graph is constructed, UMAP constructs and optimizes the layout of a low-dimensional analog to be as similar as possible. For modeling distance in low dimensions, UMAP uses a probability measure similar to Student t-distribution:

$$q_{ij} = (1 + a(y_i - y_j)^{2b})^{-1} \tag{3.4}$$

Where $a \approx 1.93$ and $b \approx 0.79$ for default UMAP.

UMAP uses binary cross-entropy (CE) as a cost function due to its capability of capturing the global data structure:

$$CE(P,Q) = \sum_i \sum_j [p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}})] \tag{3.5}$$

P is the probabilistic similarity of the high dimensional data points, and Q is for the low dimensional data points.

The derivative of the cross-entropy is used to update the coordination of the low-dimensional data points to optimize the projection space until the convergence. UMAP applied Stochastic Gradient Descent (SGD) due to its faster convergence, and it reduces memory consumption since we compute the gradients for a subset of the data set.

UMAP has several important hyper-parameters that influence its performance. These hyper-parameters are:

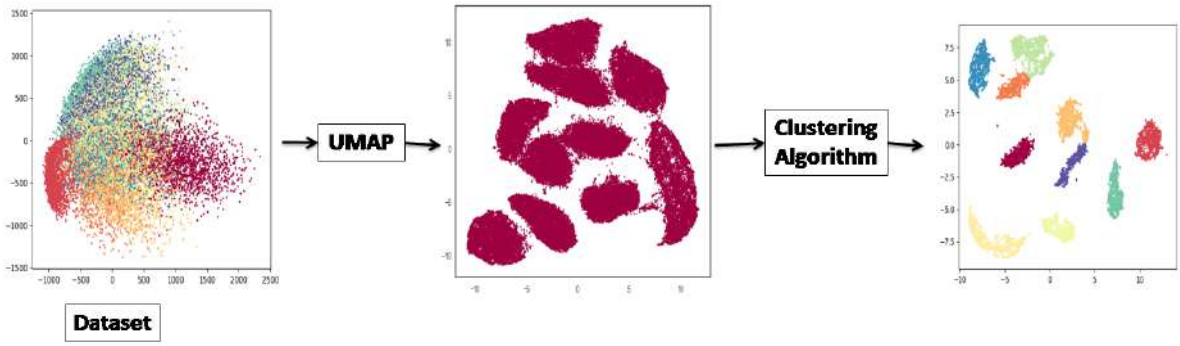

Figure 3.1 – Application of clustering algorithms on the embedding space of UMAP.

— The dimensionality of the target embedding

— The number of neighbors k, choosing a small value means the interpretation will be very local and capture fine detail structure. While choosing a considerable value means the estimation will be based on more significant regions and, thus, will miss some fine detail structure.

— The minimum allowed distance between points in the embedding space. Lower values of this minimum distance will more accurately capture the proper manifold structure but may lead to dense clouds that make visualization difficult.

3.3 Proposed Method

Our method relies primarily on applying clustering algorithms on embedding manifold extracted by manifold learning methods UMAP [12]. We described the UMAP algorithm in section 3.2. We chose four well-known clustering algorithms, which are k-means [6], HDBSCAN [32], GMM [29] and Agglomerative Clustering [30]. We will show that by augmenting the clustering task with a manifold learning technique that explicitly takes local structure into account. We can increase the quality of the clustering performance of the different algorithms. Fig. 3.1 represents the architecture of our method.

3.4 Experiments

To assess the improvement of using UMAP with the clustering algorithms studied, we conduct experiments on various diverse datasets, including standard datasets widely used to evaluate clustering algorithms.

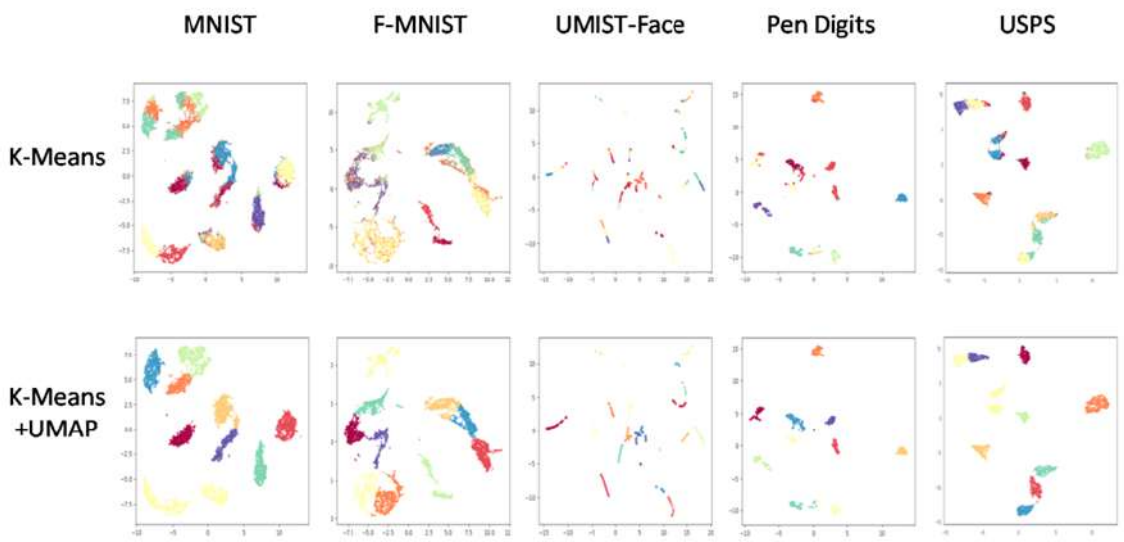

Figure 3.2 – Visualization of K-Means applied to all five datasets. The first row represents the K-Means visualization of the five datasets themselves, and the second row represents the visualization of K-Means on the UMAP embedded manifold of these datasets.

3.4.1 Requirements

We conducted our experiments on five image datasets, which are MNIST [55], Fashion MNIST [68], USPS [54], Pen Digits [69] and UMIST Face Cropped [70]. Table 2.1 summarizes the main characteristics of each dataset. To validate the performance of the clustering algorithms, we use the two standard evaluation metrics, accuracy (ACC) and Normalized Mutual Information (NMI). Please see Sec. 2.5.

3.4.2 Results

Fig. 3.2 shows the resulting clusters using k-means for visualization purposes. We could see that the visualization is better when we apply the algorithm on the UMAP embedded manifold of the five datasets. However, to better understand the effectiveness of our method at clustering, we will study each clustering algorithm by measuring its results on the different datasets using the accuracy and NMI, as well as when we apply it to the extracted features by UMAP.

Table 3.1 and Table 3.2 show the accuracy and NMI results for the clustering algorithms on five different datasets comparable to the same algorithms applied on the embedding manifold of the datasets extracted by UMAP. In both tables, improvement score rows represent the difference between the results of the algorithms and the results

Table 3.1 – Comparison between the different clustering algorithms on the five datasets according to the accuracy measure.

	MNIST	F-MNIST	UMIST Face	Pen digits	USPS
K-means	0.5278	0.4750	0.4348	0.7028	0.6678
UMAP + K-means	**0.9054**	**0.5865**	**0.7409**	**0.8843**	**0.8105**
Improvement Score	0.3776	0.1115	0.3061	0.1815	0.1427
Agglomerative	0.5751	0.5766	0.4539	0.7451	0.6834
UMAP + Agglomerative	**0.8918**	**0.5925**	**0.7270**	**0.8737**	**0.9584**
Improvement Score	0.3167	0.0159	0.2731	0.1286	0.2740
HDBSCAN	0.2765	0.2140	0.4904	0.5453	0.3529
UMAP + HDBSCAN	**0.7765**	**0.3458**	**0.6730**	**0.9004**	**0.9553**
Improvement Score	0.5000	0.1318	0.1826	0.3551	0.6024
GMM	0.4507	0.4579	0.3826	0.4836	0.4802
UMAP+GMM	**0.9159**	**0.5885**	**0.7287**	**0.8748**	**0.6727**
Improvement Score	0.4652	0.1306	0.3461	0.3912	0.1925

after applying these algorithms to the features extracted by UMAP. By doing so, we can see how UMAP can help the four clustering algorithms and how much the results have improved. The algorithms achieved great results on embedded data points, where an increase of up to 60% improves the accuracy and the range of 5% to 48% in terms of NMI. What is striking is how UMAP helped HDBSCAN to improve its result by 60 percentage points on the USPS dataset. Also, it improved better than the other algorithms in 2 of the 5 datasets with at least 50% in terms of accuracy and over 38% in terms of NMI measure. GMM is improved better than the other, on 3 of the 5 datasets, with a percentage over 34% in accuracy and over 25% in NMI measure.

The accuracy and NMI measures showed us that the studied clustering algorithms in general and HDBSCAN, as a particular case, had terrible results and especially in MNIST and Fashion MNIST datasets. The problem here is all the clustering algorithms tend to suffer from the curse of dimensionality: high dimensional data requires more observed samples to produce much density. If we could reduce the dimensionality of the data more, we would make the density more evident and make it far easier for those algorithms to cluster the data. We need manifold solid learning, where UMAP can come into play. One of the reasons which help the studied algorithms to perform well on the learned manifold is to set the min distance (the hyper-parameter of UMAP) to be 0. And thus making the

Table 3.2 – Comparison between the different clustering algorithms on the five datasets according to the NMI measure

	MNIST	F-MNIST	UMIST Face	Pen digits	USPS
K-means	0.4774	0.5139	0.6647	0.6998	0.6266
UMAP + K-means	**0.8494**	**0.6377**	**0.8663**	**0.8545**	**0.8602**
Improvement Score	0.3720	0.1238	0.2016	0.1547	0.2336
Agglomerative	0.6360	0.6080	0.6673	0.7965	0.7250
UMAP + Agglomerative	**0.8463**	**0.6511**	**0.8764**	**0.8456**	**0.9000**
Improvement Score	0.2103	0.0431	0.2091	0.0491	0.1750
HDBSCAN	0.3674	0.2535	0.6933	0.5804	0.4442
UMAP + HDBSCAN	**0.8315**	**0.6323**	**0.8427**	**0.8871**	**0.8923**
Improvement Score	0.4641	0.3788	0.1494	0.3067	0.4481
GMM	0.3882	0.5471	0.6160	0.5203	0.4232
UMAP+GMM	**0.8654**	**0.6424**	**0.8648**	**0.8447**	**0.8231**
Improvement Score	0.4772	0.0953	0.2488	0.3244	0.3999

Table 3.3 – The execution time before and after applying UMAP on the different clustering algorithms on the five datasets.

Time in second	MNIST	F-MNIST	UMIST Face	Pen digits	USPS
K-means	112.13	74.69	17.24	0.94	12.93
UMAP + K-means	**1.22**	**1.20**	**0.33**	**0.26**	**0.57**
Agglomerative	710.08	674.14	6.57	0.48	47.93
UMAP + Agglomerative	**88.31**	**100.14**	**0.03**	**0.28**	**8.51**
HDBSCAN	1603.26	1660.25	17.77	1.14	117.56
UMAP + HDBSCAN	**5.13**	**4.49**	**0.03**	**0.12**	**0.75**
GMM	24.51	26.27	3.49	0.58	25.26
UMAP+GMM	**0.51**	**0.42**	**0.06**	**0.03**	**0.14**

points packed together densely and making cleaner separations between clusters.

Table 3.3 gives us the execution time taken for each clustering algorithm on the different datasets compared to the run-time of these algorithms applied to the embedding manifold of the five datasets. We can observe that the run-time is also improved, where it was reduced to a few seconds and sometimes to a few split seconds, and this is a good achievement for our method compared to the size of the datasets. Especially for agglomerative and HDBSCAN algorithms, the run-time of HDBSCAN is reduced from over 26 minutes until around 5 seconds in MNIST and Fashion MNIST datasets. These results demonstrate that these clustering algorithms can now handle large datasets well.

3.5 Conclusion

In this study, we investigated the use of the UMAP technique for dimensionality reduction before applying several well-known clustering algorithms to datasets. We showed that it could drastically improve the performance of the studied algorithms regarding clustering accuracy and time. Experimental results indicate that the proposed approach can improve clustering performance and make conventional clustering algorithms competitive with the current state-of-the-art clustering approaches. Experiments also validate that our method allows the clustering algorithms to deal better with larger data sets.

As we saw that embedding techniques play a significant role in improving the performance of clustering techniques, our subsequent work was to suggest a technique that does joint learning for embedding and clustering. The next chapter will present our proposed technique, Unified Embedding and Clustering (UEC).

Chapter 4

Unified Embedding and Clustering

4.1 Introduction

Recent studies tended to perform the embedding and clustering together of the data. These methods perform embedding and clustering sequentially; however, the clustering phase would corrupt the representation space induced by embedding. In contrast to these methods, The embedding and clustering loss functions are optimized simultaneously in one step by our proposed technique, Unified Embedding and Clustering (UEC). It is motivated by the fact that jointly learning the embedding and clustering manifold improves clustering quality [49, 66, 10]. To start the optimization process that boosts the quality of embedding and clustering, UEC initializes the low dimensional space and the cluster assignments.

However, this optimization process can be done differently, yielding different versions of the UEC algorithm. Here we propose three (3) different variants. In the first variant, called **Comma** variant, the algorithm alternates between optimizing the embedding loss and the clustering loss functions. In the second and the third variants, called **Plus** and **Light Plus** variants, respectively, the optimization is done in one combined step. The difference between the second and third variants lies in calculating the derivative (please see Sec. 4.4). The Light Plus variant is computationally faster than the Plus variant, but the latter is more accurate than the Light Plus variant.

The rest of this chapter is organized as follows: Section 4.2 describes the proposed UEC algorithm, and Sec. 4.3 give more details about the ingredients of the algorithm. Section 4.4 presents the details of the UEC's optimization variants. Section 4.5 discusses the experimental results before concluding the chapter in Sec. 4.6.

4.2 Unification of Manifold Embedding and Clustering

Before discussing the details of the proposed method, the list of symbols to be used in the rest of this chapter is introduced in Tab. 4.1.

Table 4.1 – Description of all used symbols

Symbols	Description
Y	Input datapoints: $Y = \{y\}_{i=1}^{n}, y_i \in R^D$
Z	The representation of the data in the embedding space: $Z = \{z\}_{i=1}^{n}, z_i \in R^d$
M	The set of cluster centers: $M = \{\mu\}_{k=1}^{C}$, where C is the number of clusters
Γ_i	Local neighborhood of y_i datapoint
$p_{i\|j}$	Probability distribution between the input datapoints y_i and their $j-th$ nearest neighbor. These form the matrix: $P = \{p_{ij}\}$ to be defined later
q_{ij}	The probability distribution between the embedded datapoint z_i and its $j-th$ nearest neighbor: $Q = \{q_{ij}\}$
S_{ik}	The soft assignment distribution that indicates the probability between the points z_i and the cluster center μ_k: $S = \{s_{ik}\}$
T_{ik}	Auxiliary target distribution: $T = \{T_{ik}\}$
CE	Binary cross entropy representing the adopted embedding loss function
KL	Kullback–Leibler divergence representing the adopted clustering objective function
F	Total loss as a Weighted sum function of both CE and KL

The proposed algorithm, UEC, aims to preserve the closeness in the input space when mapping the data into the output space. Thus, data points with similar characteristics are projected nearby, and dissimilar ones are mapped apart. Like general manifold embedding, UEC considers the original data as a high-dimensional graph to be transformed into a lower-dimensional one.

UEC is about mapping high D-dimensional input data onto a d-dimensional embedded space while considering clustering constraints, where $d << D$. To achieve that, UEC optimizes the following objective function:

$$F = \alpha\ CE \oplus \beta\ KL \tag{4.1}$$

Algorithm 1 UEC

Input: Dataset $Y = \{y_i\}_{i=1}^{n}$, number of cluster C, dimension of the embedding space d, min-dist, number of neighbors nn
Output: Embedded dataset $Z = \{z_i\}_{i=1}^{n}$, set of centers $M = \{\mu_k\}_{k=1}^{C}$.

1: Construct the graph of the original data
2: Compute the affinity matrix for the input data Y Eq. 4.3 and 4.5.
3: Initialize the embedding space of dimension d.
4: Initialize the cluster centers.
5: **while** The convergence criterion is not met **do**
6: Compute the affinity matrix for the embedded data Z, Eq. 4.6.
7: Compute the soft assignment of the embedded data to the clusters $\mathcal{S}(Z, M)$, Eq. 4.7.
8: Compute the probabilistic point-to-cluster assignments using the obtained soft assignment $\mathcal{T}(Z, M)$, Eq. 4.8.
9: Update the coordinates of the embedded data Z, as in Eq. 4.14.
10: Update the centers, as in Eq. 4.12.
11: **end while**

The first term (CE) is the cross-entropy that assesses the embedding quality in the low-dimensional space. It is referred to as *the embedding loss function*. The second term (KL) is the Kullback-Leibler divergence, which assesses the clustering quality. It is referred to as *the clustering loss function*. The quantities α and β define the relative weights of the overall function's two-loss component and control the trade-off between embedding and clustering.

We can optimize this function in two different ways. Each of them follows an interpretation of the $\oplus$ symbol:

1. $\oplus =$ ',' (comma) UEC takes one gradient step for the embedding optimization problem, passes the new parameters to the clustering optimization problem, takes one gradient step for the second optimization problem, then passes the estimated parameters to the first optimization problem again, and repeats this alternating sequence. This scenario is referred to as a *sequential update*.

2. $\oplus =$ '+' (plus) that indicates that the evaluation of Eq. 4.1 is done in one combined step (multi-objective function) to update the sought quantities simultaneously. This scenario is referred to as *joint update*.

Equation 4.1 depends on several quantities (matrices), namely P, Q, T and S described in Tab. 4.1. The matrix P forms the affinity scores between the individual data

points and their nearest neighbors in the input space. The matrix, Q, represents the similarity between the embedding space's data points. The matrix S is a soft assignment of embedded data to clusters. At the same time, T represents the probabilistic point-to-cluster assignments using the obtained soft assignment S (The formal definition of these matrices will follow below). Equation 4.1 can be rewritten as follows:

$$F(P,Q,T,S) = \alpha\ CE(P,Q) \oplus \beta\ KL(T||S) \tag{4.2}$$

CE computes the total entropy between P and Q, while KL measures the relative entropy (divergence) between S and T. This objective function will be used to analytically compute the coordinates of the set Z (coordinates of the data in the embedded space) and the centers M.

Before delving into the details, it is worthwhile to portray the structure of the UEC algorithm.

4.3 Formulation of the algorithm ingredients

4.3.1 Computation of the affinity matrix P

The affinity matrix P represents the similarity scores between pairs of data points using the exponential probability:

$$p_{i|j} = \exp(-\frac{d(y_i,y_j) - \rho_i}{\sigma_i}) \tag{4.3}$$

Given an input hyper-parameter nn which represents the number of neighbors, $d(y_i,y_j)$ is the distance between the i^{th} datapoint and its j^{th} nearest neighbor. ρ_i is defined by:

$$\rho_i = min\{d(y_i,y_j) \mid 1 \leq j \leq nn, d(y_i,y_j) > 0\} \tag{4.4}$$

Which represents the distance between i^{th} datapoint and its first nearest neighbor. It ensures the local connectivity of the manifold. For each y_i, σ_i is defined by:

$$\sum_{j=1}^{nn} exp\left(\frac{-max(0, d(y_i,y_{i_j}) - \rho_i)}{\sigma_i}\right) = log_2(nn)$$

We use a symmetrization of the high-dimensional probability since the weight of the edge between the i^{th} and j^{th} nodes is not equal to the weight between j^{th} and i^{th} nodes. The final formulation of P is given as follows:

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i} \tag{4.5}$$

We utilize the Nearest-Neighbor-Descent (k-NN) algorithm [71] to represent the data's underlying structure and construct local neighborhood Γ of point y_i. Therefore local neighborhood is $\Gamma_i = \{y_j | y_j \in knn(y_i), y_i \in knn(y_j)\}$. However, the resulting k-NN graph often contains disconnected components and potentially isolated vertices. Isolated vertices violate the assumption that the underlying manifold is locally connected, and disconnected components negatively impact the initialization of the low dimensional space. Inspired from [72], we refine the graph construction and avoid the problem of the isolated vertices by increasing the connectivity of the k-NN graph using a maximum spanning tree (MST) [73].

4.3.2 Initialization

We use a spectral layout [74] to initialize the embedding. To map the data into a low-dimensional space, the spectral embedding method computes the eigenvectors of the affinity matrix of the graph. This provides both faster convergence and greater stability of the algorithm. To initialize the centers of the clusters, we use a centroid-based algorithm (e.g., k-means, GMM, etc.).

4.3.3 Computation of the affinity Matrix Q

The affinity matrix Q represents the similarity scores between each embedded data-point and its neighbors. It is computed using a smooth approximation of the membership strength:

$$q_{ij} = (1 + a\left\|z_i - z_j\right\|_2^{2b})^{-1} \tag{4.6}$$

The parameters a and b are defined using the piece-wise non-linear least-square fitting $\psi : \mathbb{R}_d \times \mathbb{R}_d \to [0, 1]$ where

$$\psi(z_i, z_j) = \begin{cases} 1 & if \left\| z_i - z_j \right\|_2 \leq \text{min-dist} \\ \exp(-(\left\| z_i - z_j \right\|_2 - \text{min-dist})) & \text{otherwise} \end{cases}$$

where $min-dist$ is a hyper-parameter representing the desired separation between close points in the embedding space.

4.3.4 Computation of the soft assignments S

The matrix S is computed using a smooth approximation of the membership strength between the embedded points z_i and the cluster centers μ_k as follows:

$$S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1} \tag{4.7}$$

Where a and b are the same ones used in Eq. 4.6. We could also consider the following form like in DEC [7] (which is inspired from t-SNE [43]):

$$S_{ik} = \frac{(1 + a\|z_i - \mu_k\|_2^{2b})^{-1}}{\sum_k (1 + a\|z_i - \mu_k\|_2^{2b})^{-1}}$$

However, it has been shown in UMAP [12] that the normalization increases processing time without improving accuracy.

4.3.5 Computation of the auxiliary target distribution T

The matrix T should satisfy three constraints: (1) improving the cluster purity, (2) ensuring high-confidence assignments to get more emphasis, and (3) preventing large clusters from distorting the embedding space by normalizing the loss contribution of each center. A formulation that addresses these constraints is given as follows:

$$T_{ik} = \frac{S_{ik} / \sum_l S_{lk}}{\sum_m (S_{lm} / \sum_l S_{lm})} \tag{4.8}$$

The T_{ik} numerator ensures that the first condition is met. Normalizing the membership of a data point by the sum of the memberships of other data points in the same cluster makes this point have a hard assignment (closer to 0 or 1). The denominator guarantees

the third constraint, which enforces our preference for having balanced assignments and preventing large clusters from distorting the embedding space. Points that have hard assignments are involved in the process of finding the assignment of stray points.

4.3.6 Formulation of the optimization problem

The coordinates of the data points and the centers of the clusters are updated in light of the minimization of the two objective functions: the embedding and the clustering loss functions. These functions are coupled in Eq. 4.2. Before discussing the two optimization options (see Sec. 4.2) presented earlier, we formulate the first term, which is the embedding objective loss represented as binary cross-entropy (CE). CE is given as follows:

$$CE(P,Q) = \sum_i \sum_j [p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}})] \tag{4.9}$$

where P and Q are the probabilistic similarity scores of the input data Y and that of the embedded data Z respectively. The second term is the clustering objective function, which is defined as the Kullback–Leibler (KL) divergence function between the soft assignments S and the auxiliary distribution T:

$$KL(T||S) = \sum_i \sum_k T_{ik} \log \frac{T_{ik}}{S_{ik}} \tag{4.10}$$

Due to the relative entropy of the KL divergence function and the ability to learn from high-confidence assignments and auxiliary target distribution, it is used to refine clusters. The combination of cross entropy and KL divergence loss functions makes our algorithm capable of capturing the local and global data structure. The objective function, terms, and all quantities used have been defined. We discuss the optimization problem to update the coordinates of Z and the clusters' centers.

4.4 Optimization of the objective function

The coordinates of a datapoint z_i and centers of clusters μ_j are updated at each time step t until the criterion convergence parameter is met. Due to its rapid convergence and low memory consumption, we use Stochastic Gradient Descent (SGD) as an optimization

algorithm. As illustrated in Alg. 1, the optimization steps are repeated until the change in the cluster assignment of the points is less than a threshold $1e-5$.

As indicated earlier, the update of z_i and μ_j can occur according to two variants.

4.4.1 Comma (Sequential) variant

In the comma variant, the terms of the objective function are sequentially evaluated.

4.4.1.1 Update of the embedding

The data coordinates in the embedded space will be updated twice in a sequential manner. The first update stems from the embedding loss and is given as follows:

$$z_i(t+1) = z_i(t) - \eta \frac{\delta CE}{\delta z_i}$$

where η is a learning rate. The quantity $\frac{\delta CE}{\delta z_i}$ is expressed as follows:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/(a\left\|z_i - z_j\right\|_2^{2(b-1)}) + \left\|z_i - z_j\right\|_2^2} - \frac{2b(1-p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1 + a\left\|z_i - z_j\right\|_2^{2b})} \right] \left\|z_i - z_j\right\| \tag{4.11}$$

For more details, see Appendix A. While the second update of the coordinates of z_i comes from the clustering loss.

$$z_i(t+1) = z_i(t) - \eta \frac{\delta KL}{\delta z_i}$$

The partial derivative of the clustering loss function (KL) given by Eq. 4.10 w.r.t. z_i reads as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}}$$

4.4.1.2 Update of the cluster centers

The centers μ_k do not depend on the embedding loss function. Hence, the centers of the clusters are updated using the derivative of the clustering loss function, the KL-

divergence (Eq. 6.9), as follows:

$$\mu_k(t+1) = \mu_k(t) - \eta \frac{\delta KL}{\delta \mu_k} \tag{4.12}$$

where η is a learning rate, and $\frac{\delta KL}{\delta \mu_k}$ is as follow:

$$\frac{\delta KL}{\delta \mu_k} = \sum_i -\frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{4.13}$$

4.4.2 Plus (Combined) variant

The main difference between the combined and the sequential variants is that the clustering influences the computation of z_i and μ_j in the former.

4.4.2.1 Update of the embedding

According to this second variant, the coordinates of the datapoints z_i are updated using the embedding and the clustering loss functions simultaneously.

From Eq. 4.1, the update is executed as follows:

$$z_i(t+1) = z_i(t) - \eta \frac{\partial F}{\partial z_i} \tag{4.14}$$

$$= z_i(t) - \eta \left(\alpha \frac{\delta CE}{\delta z_i} + \beta \frac{\delta KL}{\delta z_i} \right) \tag{4.15}$$

Where $\frac{\delta CE}{\delta z_i}$ is given by Eq. 4.11 and $\frac{\delta KL}{\delta z_i}$ is given as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i} (1 + \log \frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i} (1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \tag{4.16}$$

The derivative of KL is the sum of 2 terms since the derivation of T_{ik} w.r.t. z_i is computed according to two cases: 1) When $i' = i$, 2) When $i' \neq i$ (see Appendix B for more details).

The final formulation of the update is obtained by combining Eq. 4.11 and Eq. 4.16:

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \tag{4.17}$$

4.4.2.2 Update of the cluster centers

Since CE does not depend on μ_k, only the clustering loss function, KL is relevant:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{2ab \|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} (1 + \log \frac{T_{ik'}}{S_{ik'}}) \right]$$

The derivative of the KL divergence is the sum of two parts requiring the consideration of two cases: 1) When $k' = k$, 2) When $k' \neq k$. For more detail, see Appendix B.

4.4.3 Light Plus variant

In the *combined variant*, we consider that the update of z_i and μ_k depends on the auxiliary target distribution, T. While this variant significantly improves the proposed UEC's performance, it is not highly efficient in terms of computational time due to the heavy computation involved in the gradient descent update, hence this light version of the *combined variant*. The underlying assumption of this third variant is to consider T_{ik} not dependent on z_i and μ_k. More details are provided in Appendix C.

4.4.3.1 Update of the embedding

KL divergence is derived w.r.t. z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab \|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{4.18}$$

By substituting the derivative of CE (Eq. 4.11) and the derivative of KL divergence (Eq. 4.18) in the total loss function (Eq. 4.17), we then obtain the final update of $z_i(t+1)$ as shown in Eq. 4.15.

4.4.3.2 Update of the cluster centers

The derivative of total loss F (Eq. 4.1) w.r.t. μ_k is computed only for the KL-divergence function, which depends on the cluster centroids. The formulation obtained in the sequential variant derivations will be applied here for updating the centers (See Eq. 4.12 and Eq. 4.13).

4.4.4 Implementation

Following the work of [12] and [71], the practical implementation of this algorithm requires k-nearest neighbor calculation and efficient optimization via stochastic gradient descent. As mentioned above, we used the Nearest Neighbor Descent (NND) algorithm of [71] to obtain the optimal number of the nearest neighbors. The NND paper reported that the algorithm's empirical complexity is $O(N^{1.14})$. In optimization of the embedding under the provided objective function, we follow the work of [12]. This gives a very efficient approximate stochastic gradient descent algorithm since there is no normalization requirement. From what was mentioned above, the overall complexity is empirically approximately $O(N^{1.14})$.

4.5 Experiments and discussion

This section will show the performance of the proposed 3 variants of UEC on a set of benchmarks. Specifically, we will discuss the following experiments:

— The first experiment evaluates the three variants and compare their performance in term of external validation, internal validation, and computational time; see Sec. 4.5.2.

— In the second experiment, we study the effect of the number of clusters and the initialization of the clusters' centers on the performance of UEC variants, see Sec. 4.5.3.

— In the third experiment, we study the sensitivity of UEC towards the initialization of embedding space, see Sec. 4.5.4.

— In the fourth experiment, we assess the ability of UEC to Preserve the Local and Global structure of the data, see Sec. 4.5.5.

— In the fifth experiment, we discuss the parameters α and β in Eq. 4.1 and their effect, see Sec. 4.5.6.

— In the sixth experiment, we compare the performance of UEC variants against the state-of-the-art manifold and deep clustering methods, see Sec. 4.5.7.

— In the seventh experiment, we evaluate the performance of UEC variants challenging datasets, see Sec. 4.5.8.

— Qualitative results are presented in the last section, see Sec. 4.5.9

4.5.1 Requirements

4.5.1.1 Datasets

We conduct experiments on six benchmark datasets given as follows: IRIS, Spiral, USPS [54], MNIST [55], CIFAR-10 [56], Reuters [57]. We evaluate all clustering methods using two kinds of measures: External and Internal validation metrics. We used as External validation metrics the ACCuracy (ACC) [59] and Normalized Mutual Information (NMI). We use three well-known internal validation indices, which are Davis-Bouldin (DB)[62], Silhouette coefficient (S) [61], and Calinski-Harabaz (CH) [60]. More details about those datasets and evaluation metrics are given in Sec. 2.5

Implementation Details

In the optimization step of UEC, our weighted sum function is minimized using Stochastic Gradient Descent (SDG). We set the learning rate to 1.0 and then decreased it by 10 every 10 epoch. The number of epochs is set to 200 for big datasets and 500 for small ones.

4.5.2 Comparison of the variants

In this experiment, We evaluate the performance of each variant of the UEC (*comma variant*, *plus variant*, *light plus variant*) in terms of three aspects: (1) External validation, (2) internal validation, and (3) computational time.

As an external validation measure, we used the accuracy and NMI measures to evaluate the performance of UEC's three variants. Through Tab. 4.2, We can see that the

clustering performance of *Plus variant* is better than the other two versions. However, Table 4.3 shows that *Comma variant* and *Light plus variant* are better than *Plus variant* in the execution time as expected given its high computational requirements involving several equations as shown in Appendix B. *Comma variant* and *Light plus variant* are approximately near to each other in the execution time. However, *Light plus variant* is better than *Comma variant* in the clustering performance. Therefore, these results support our claim that the joint optimization of embedding and the clustering loss functions improves the clustering performance of our algorithm.

Internal validation intends to quantify clustering quality, usually using two criteria: *Compactness* and *Separation*. The first criterion measures the data objects' similarity in the individual clusters. The second criterion measures how distinct or well-separated clusters are from each other. Often, these two criteria are embedded in various clustering quality indices. In this study, we use three internal validation indices, which are Davis-Bouldin (DB) [62], Silhouette coefficient (S) [61], and Calinski-Harabaz (CH) [60].

Tables 4.4, 4.5, and 4.6 show the clustering quality scores of the UEC variants. They indicate that the *Plus variant* is the best among the three variants across the three indices. The UEC variants obtain approximately the same DB and S results on MNIST and USPS, presumably because of the similar nature of these two datasets. The results on CIFAR-10 are less competitive, maybe because of the poor quality of the images leading to cluster overlapping.

On the other hand, the three variants perform well on the Reuters dataset using each of the three indices. In general, the UEC variants can preserve the between and within-cluster distances because they are designed to preserve the global and local structures of the data.

Remark: *Based on the results achieved by the three variants of UEC, we use only the Light plus variant in the upcoming experiments (in Sec. 4.5.3 and Sec. 4.5.6), since Light plus variant is the fastest one among the three versions. Its clustering performance is near to Plus variant.*

Table 4.2 – Evaluating the performance of the UEC's three variants in Accuracy (ACC) and NMI.

Models	Dataset			
	USPS		MNIST	
	ACC	NMI	ACC	NMI
Comma variant	0.967	0.928	0.959	0.932
Plus variant	0.982	0.948	0.988	0.956
Light plus variant	0.979	0.937	0.986	0.950

Table 4.3 – Evaluating the performance of the UEC's three variants in terms of Execution time (in seconds).

Models	Dataset	
	USPS	MNIST
Comma variant	70	150
Plus variant	180	1226
Light plus variant	47	115

4.5.3 Effect of the number of clusters

In this experiment, we study two aspects. First, We analyze the sensitivity of the UEC algorithm toward changing the number of clusters and then study the effect of different initialization of the clusters' centers.

To study The effect of changing the number of clusters on the performance of UEC on three datasets: USPS, MNIST, and CIFAR-10. We use the Average Silhouette method to evaluate the performance of UEC for different values of the number of clusters k in a range of $[2, 30]$. According to the Silhouette method, we observe that the best number of clusters for the three datasets is 10; see Fig. 4.1. We made one more step to be sure that 10 is the optimal number of clusters for the three datasets. We evaluate UEC on the same range using Accuracy and NMI metrics. Figure 4.2 shows that the best performance of UEC on the three datasets is 10 clusters. We can observe that the performance of UEC is coherent and consistent in both experiments.

The second part of this experiment is to study the sensitivity of the UEC algorithm toward the initialization of the clusters' centers. We perform three types of initialization using k-means, Gaussian mixture models (GMM), and randomly chosen medoids. The experiments are conducted using *Light plus variant* on the considered datasets (see Sec. 4.5.2). Table 4.7 shows our algorithm's performance using the three initializations

Table 4.4 – Clustering quality using the Davies-Bouldin index.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	1.676	1.593	5.148	0.701
Plus variant	1.581	1.494	4.264	0.559
Light plus variant	1.634	1.555	4.721	0.623

Table 4.5 – Clustering quality using the Silhouette index.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	0.201	0.219	-0.001	0.315
Plus variant	0.278	0.282	0.009	0.495
Light plus variant	0.222	0.231	0.006	0.417

evaluated by the accuracy measure. We can see that the UEC algorithm is not sensitive to the type of initialization adopted (even with the random centroid initialization, it achieves good results).

Notice: *Based on the achieved results, in the upcoming experiments, we use k-means to initialize the centers.*

4.5.4 Embedding space Initialization

In this experiment, we analyze the sensitivity of the UEC algorithm toward the initialization of the embedding space. We perform two types of initialization using spectral embedding algorithm and random initialization. The experiments are conducted using *Light plus variant* on the considered datasets. Table 4.8 shows the performance of our algorithm using both initializations evaluated by the accuracy measure. We can see that

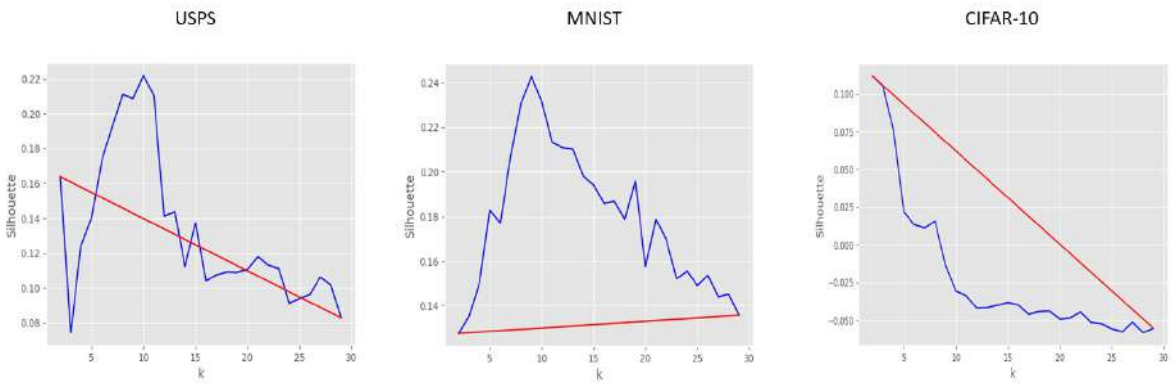

Figure 4.1 – The effect of changing the number of clusters measured by Silhouette score.

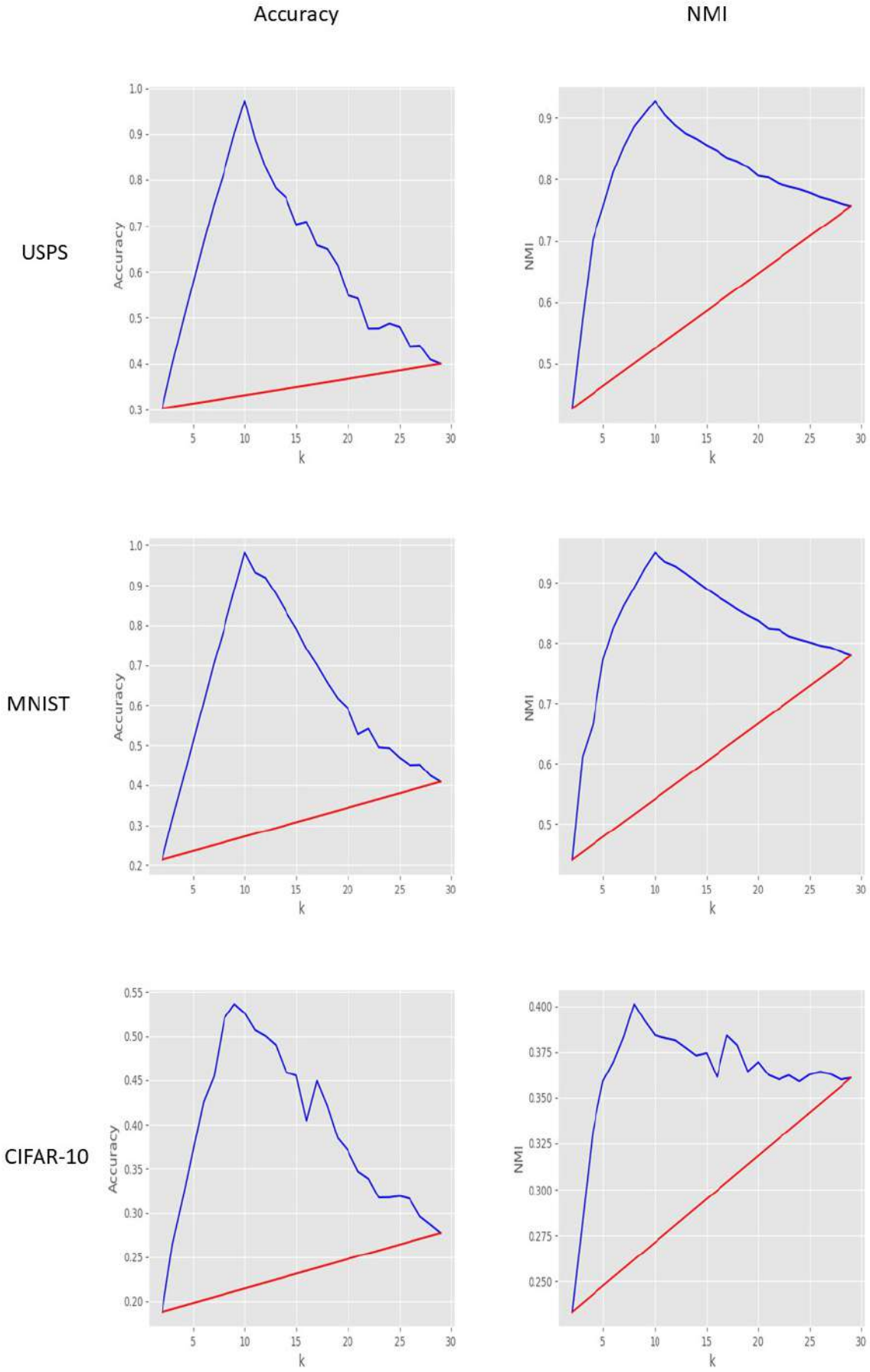

Figure 4.2 – The effect of changing the number of clusters measured by Accuracy and NMI.

Table 4.6 – Clustering quality using the Calinski-Harabaz index.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Comma variant	1123.754	9682.256	997.84	16481.130
Plus variant	1398.425	10189.580	1289.475	20142.254
Light plus variant	1211.068	9819.213	1179.962	18248.961

Table 4.7 – Effect of center initialization on the performance.

Algorithm initialization	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
random	0.975 ± 0.004	0.981 ± 0.005	0.515 ± 0.01	0.955 ± 0.002
K-means	0.979 ± 0.002	0.986 ± 0.003	0.526 ± 0.002	0.962 ± 0.001
GMM	0.980 ± 0.003	0.985 ± 0.002	0.524 ± 0.002	0.961 ± 0.002

the UEC algorithm is not sensitive to the type of initialization adopted (even with random initialization, it achieves good results).

Table 4.8 – Effect of the embedding space initialization on the performance.

Algorithm initialization	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
Random initialization	0.976 ± 0.03	0.984 ± 0.02	0.520 ± 0.04	0.949 ± 0.01
Spectral embedding	0.979 ± 0.02	0.986 ± 0.01	0.526 ± 0.02	0.962 ± 0.005

4.5.5 Preserving the local and the global structures of the data

In the first part of this experiment, we assess the ability of k-NN graphs to capture the local structure and study the effect of the number of neighbors nn. In addition, this experiment is important to decide the best value of nn for the upcoming experiments. To do so, we conduct an extensive sensitivity analysis experiment, varying the values of nn in a range $[2, 100]$ and evaluating the performance of UEC using three measures: two are internal validation metrics, namely Davis-Bouldin and Silhouette coefficient and the third one is an external metric which is the accuracy. The Internal metrics assess how well the data points within the same cluster are similar to each other and how distinct different clusters are from each other. Internal measures indicate how well the local structure of data is preserved. The external metrics provide insights into the extent to which obtained clusters fit known or expected patterns so that external measures are a good indicator of how well the global structure of the data is maintained.

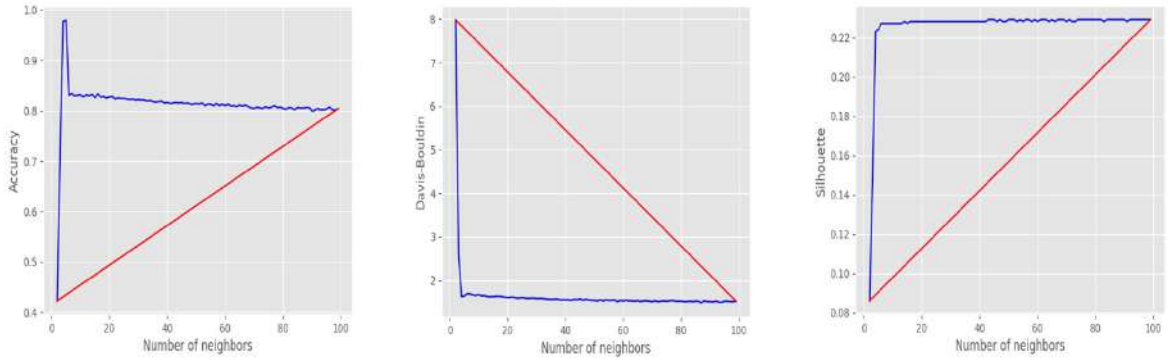

Figure 4.3 – Preservation of the local structure using k-NN graph.

In this experiment, we consider the USPS dataset to obtain the results shown in Fig. 4.3. The best performance in terms of accuracy is obtained when the number of neighbors is set to 4 and 5. The internal measures, on the other hand, in the last two sub-figures show that the local structure is well captured since nn after a certain threshold value, 4, does not affect the internal measures becoming stable.

In the upcoming experiments, we fixed nn for the small datasets to be 5 and for the large datasets to be 30 as the default value.

In the second part of this experiment, we will study how the cross entropy (CE) function can maintain the global structure of the data. First, let us observe the following cases to understand the behavior of CE:

— If the distances between points in both the high and low-dimensional space are small, then CE must be low or non-existent.

— If the distances between points in both the high and low-dimensional space are large, then CE must be low or non-existent.

— CE must be large if the distances between points in the high-dimensional space are small (resp. large) and the distances in the low-dimensional space are large (resp. small).

On the other hand, from the definition of CE, Eq. 4.9:

$$CE(P,Q) = \sum_i \sum_j [P(X)\log(\frac{P(X)}{Q(Y)}) + (1-P(X))\log(\frac{1-P(X)}{1-Q(Y)})]$$

where X is the distance between the data points in the high-dimensional space and Y is

the distance between the data points in the low-dimensional space.

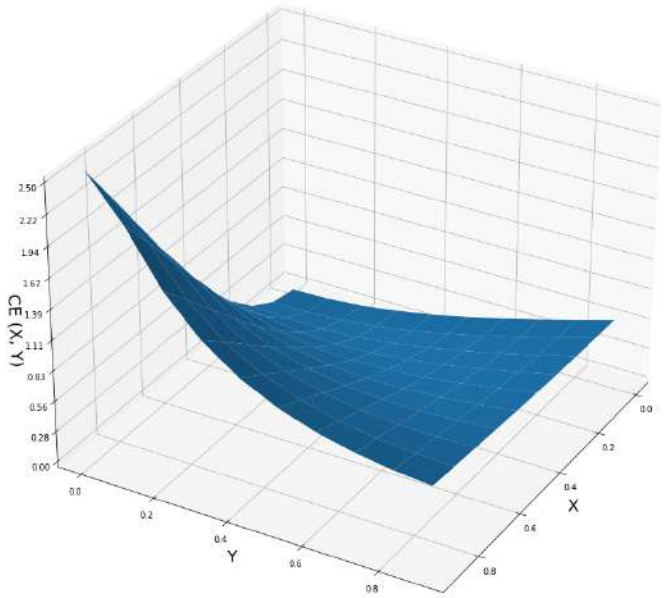

Figure 4.4 – Preservation of the global structure using CE function.

Considering the USPS dataset, X and Y are plotted in Fig. 4.4 along with the CE loss between them. Moreover, when the X distances are large, and the Y distances are small, CE is large (meaning that far apart points in the high dimensional projected far to each other in the low dimensional space). In conclusion, the $CE(X,Y)$ function is able to preserve both local and global distances.

4.5.6 Effect of Alpha and Beta

Recall that α is related to the embedding loss, while β is related to clustering. To study the effect of the parameters α and β on the loss function, we vary them in the unit interval $[0,1]$ and observe their effect on the UEC's performance and execution time. The experiments are conducted using *Light plus variant* following the remark in Sec. 4.5.2. To perform such analysis, two experiments are designed as follows:

Experiment 1: we study the effect of the parameters on the gradient magnitude order of both embedding and clustering loss functions w.r.t. z_i (more details about the magnitude of both gradients are presented in D) by varying α and β in $[0, 1]$. Using the USPS dataset, the accuracy and NMI results are presented in Tab. 4.9 and Tab. 4.10, respectively. For computational reasons, we have considered only USPS due to its size. The algorithm achieves better results when α and β both go to 1 and worst results when they go to 0 (since only the centroids move). If we allow data points to move, the model makes a quantum leap in its performance as we see in Tab. 4.9 comparing the outcome when the parameters are set to 0 and 0.2, the accuracy increases from 0.615 to 0.951. The performance improves as β increases, but the performance remains relatively constant as α increases. We can also observe that the execution time increases when α decreases (because the algorithm needs more epochs to converge).

Table 4.9 – Effect of α and β on the performance of the algorithm measured by accuracy.

β \ α	0	0.2	0.4	0.6	0.8	1
0	0.615	0.951	0.953	0.956	0.958	0.961
0.2	0.958	0.961	0.962	0.964	0.965	0.967
0.4	0.961	0.963	0.965	0.967	0.969	0.970
0.6	0.963	0.966	0.968	0.970	0.971	0.973
0.8	0.965	0.968	0.971	0.973	0.975	0.976
1	0.972	0.973	0.974	0.976	0.977	0.979

Table 4.10 – Effect of α and β on the performance of the algorithm measured by NMI.

β \ α	0	0.2	0.4	0.6	0.8	1
0	0.575	0.904	0.908	0.912	0.916	0.919
0.2	0.910	0.913	0.916	0.919	0.921	0.923
0.4	0.914	0.917	0.920	0.923	0.925	0.927
0.6	0.918	0.921	0.923	0.926	0.928	0.930
0.8	0.921	0.924	0.927	0.929	0.932	0.934
1	0.923	0.926	0.929	0.932	0.935	0.937

Experiment 2: We study the effect of varying α and β on a modified gradient magnitude order of both embedding and clustering loss functions. This experiment aims to show if we clip the values of the gradients of the loss function, this modification could improve the algorithm's performance.

The gradient values of the embedding loss are found to be in $]-20, 20[$, while those of the clustering loss are in $]0, 20[$. However, most of the values for the gradient of embedding and clustering loss functions are in $[-4, 4]$ and $[0, 1]$, respectively. We, therefore, clip the values of the embedding loss gradient to $[-4, 4]$, and we divide the values by 4 to bring it to the interval $[-1, 1]$. The values of the clustering loss gradient are clipped to $[0, 1]$. Tables 4.11 and 4.12 show the accuracy and NMI of the algorithm after the clipping transformation. We can observe a slight positive change in the performance results.

Table 4.11 – Effect of different values of α and β on the algorithm's performance measured by the accuracy.

β \ α	0	0.2	0.4	0.6	0.8	1
0	0.594	0.944	0.948	0.951	0.954	0.957
0.2	0.956	0.957	0.958	0.960	0.961	0.962
0.4	0.960	0.962	0.963	0.964	0.965	0.966
0.6	0.963	0.966	0.967	0.968	0.969	0.970
0.8	0.967	0.968	0.970	0.971	0.972	0.973
1	0.969	0.971	0.972	0.973	0.975	0.976

Table 4.12 – Effect of different values of α and β on the performance of the algorithm measured by the NMI.

β \ α	0	0.2	0.4	0.6	0.8	1
0	0.571	0.900	0.904	0.907	0.912	0.915
0.2	0.912	0.914	0.916	0.918	0.920	0.922
0.4	0.915	0.917	0.919	0.921	0.924	0.926
0.6	0.918	0.920	0.922	0.924	0.927	0.929
0.8	0.921	0.923	0.925	0.927	0.929	0.931
1	0.924	0.926	0.928	0.930	0.932	0.933

4.5.7 Comparative Study

4.5.7.1 Baseline Methods

The proposed UEC is compared with a set of deep and manifold-based clustering methods including state-of-the-art deep clustering methods: k-means [6], deep embedded clustering (DEC) [7], Joint Unsupervised Learning (JULE) [44], Deep Embedded Regularized Clustering (DEPICT) [8], Deep Adaptive Clustering (DAC) [9], Information Maximizing Self-Augmented Training IMSAT [45]. Spectral Net [38], Deep Embedded Di-

mensionality Reduction Clustering (DERC) [10], and Dynamic Autoencoder DynAE [39]. For these methods, the performance results are taken from the original publications.

4.5.7.2 Experiment results

Tables 4.13 and 4.14 outline the performance in terms of accuracy and NMI, respectively, where the top three accuracy scores are highlighted. Table 4.13 shows that both variants of UEC (Plus and Light Plus variant) outperform the other algorithms across all benchmarks. It outperforms most of the deep clustering methods by a significant margin. Among the pros of manifold embedding techniques are they do not need any fine-tuning, in contrast to the deep clustering models, which require tweaking several hyper-parameters and fine-tuning to achieve better results. This advantage makes UEC significantly a better embedding-and-clustering choice than the other alternative clustering models. In addition, JULE, DEPICT, and DAC techniques are designed for image datasets, so these techniques cannot be performed on other datasets, such as Reuters. In contrast, UEC can be applied to any dataset.

Table 4.13 – UEC vs. other baselines: accuracy scores.

	Dataset			
Models	USPS	MNIST	CIFAR-10	Reuters
k-means [6]	0.668	0.572	0.228	0.524
DEC [7]	0.619	0.843	0.301	0.722
JULE [44]	0.950	0.964	0.271	-
IMSAT [45]	0.976	0.984	0.456	0.719
DEPICT [8]	0.964	0.965	0.342	-
DAC [9]	0.972	0.978	0.522	-
Spectral Net [38]	0.965	0.971	0.322	0.803
DERC [10]	0.977	0.975	-	-
DynAE [39]	**0.981**	**0.987**	**0.530**	**0.937**
UEC (Plus variant)	**0.982**	**0.988**	**0.534**	**0.965**
UEC (Light plus variant)	**0.979**	**0.986**	**0.526**	**0.962**

Table 4.14 – UEC vs. other baselines: NMI scores.

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
k-means	0.450	0.499	0.087	0.497
DEC	0.586	0.816	0.256	0.703
JULE	0.913	0.913	0.192	-
IMSAT	0.945	0.953	0.431	0.594
DEPICT	0.927	0.917	0.306	-
DAC	0.928	0.935	0.395	-
Spectral Net	0.914	0.924	0.288	**0.674**
DERC	**0.942**	0.927	0.316	-
DynAE	**0.948**	**0.964**	**0.403**	-
GCML	0.902	0.946	0.374	0.590
UEC (Plus variant)	**0.948**	**0.956**	**0.412**	**0.894**
UEC (Light plus variant)	0.937	**0.950**	**0.398**	**0.879**

Table 4.15 – Evaluating the performance of our three variants against the other techniques in terms of Execution time (in seconds).

Models	Dataset			
	USPS	MNIST	CIFAR-10	Reuters
k-means	12	112	152	15
DEC	53	693	1035	80
JULE	2540	12500	21250	-
IMSAT	1160	4675	9687	1856
DEPICT	1778	9561	13570	-
DAC	1690	9670	12280	-
Spectral Net	6480	11430	19430	9720
DERC	3247	10195	17400	-
DynAE	7910	10808	26270	-
GCML	8040	15000	32500	7880
UEC (Plus variant)	180	1226	3065	160
UEC (Light plus variant)	47	115	965	40

4.5.8 Evaluation on Challenging datasets

The objective of this experiment is to evaluate the performance of UEC on challenging datasets, such as spiral, IRIS, EngyTime, and Atom, compared to a set of clustering algorithms. Those datasets address specific challenges to clustering algorithms, such as lack of linear separability, different or small internal class spacing, classes defined by data density rather than data spacing, no cluster structure, outliers, or touching classes [75, 76]. Through Tab. 4.16, it can be seen we compared UEC with conventional algorithms in this table. There is no Deep Clustering (DC) algorithm because it is known that these techniques require large datasets to be able to train the DC techniques on these datasets.

Data augmentation can be done, but DC techniques still fall into the problem of over-fitting because DC techniques have large numbers of hidden features whose values are radically under-determined by small data. The ability to cluster small datasets is a plus point in favor of UEC against deep clustering algorithms. In addition, through Tab. 4.16, we can observe that UEC outperforms most of the compared algorithms with a significant margin in both datasets. In addition, we observe that the performance of UEC is competitive to HDBSCAN on the Atom dataset, which means both techniques performs well on dataset defined by data density. UEC has successfully overcome the mentioned challenges posed by these datasets.

Figure 4.5 represents 2D visualization of IRIS, Spiral, Engytime, and Atom datasets using UEC versus original visualization. We note that UEC always seeks to maintain the similarity between the points so that the original space's close points are mapped to each other in the embedding space, and the diverging points fall far from each other. We can also notice that UEC can separate overlapped clusters from each other well, as is the case in IRIS and Engytime datasets.

Table 4.16 – Evaluating the performance of UEC's variants against the other techniques in terms of Accuracy (ACC) and NMI.

Models	Dataset							
	Spiral		IRIS		EngyTime		Atom	
	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI
K-means	0.523	0.218	0.893	0.758	0.951	0.729	0.717	0.298
GMM	0.473	0.202	0.900	0.777	0.951	0.794	0.642	0.200
Spectral clustering	0.486	0.298	0.906	0.805	0.962	0.776	0.501	0.002
Agglomerative	0.570	0.219	0.893	0.770	0.923	0.646	0.657	0.219
HDBSCAN	0.406	0.129	0.906	0.713	0.575	0.385	1.0	1.0
UEC (Plus variant)	0.829	0.753	0.946	0.845	0.987	0.904	1.0	1.0
UEC (Light plus variant)	0.794	0.733	0.934	0.827	0.982	0.895	1.0	1.0

4.5.9 Qualitative results

The discriminative ability of the three variants of UEC can be illustrated in Fig. 4.6. The figure represents a 2D visualization of USPS and MNIST datasets. These representations are obtained by projecting the data onto a 2-dimension space. We can observe that in all datasets, the clusters are well-separated.

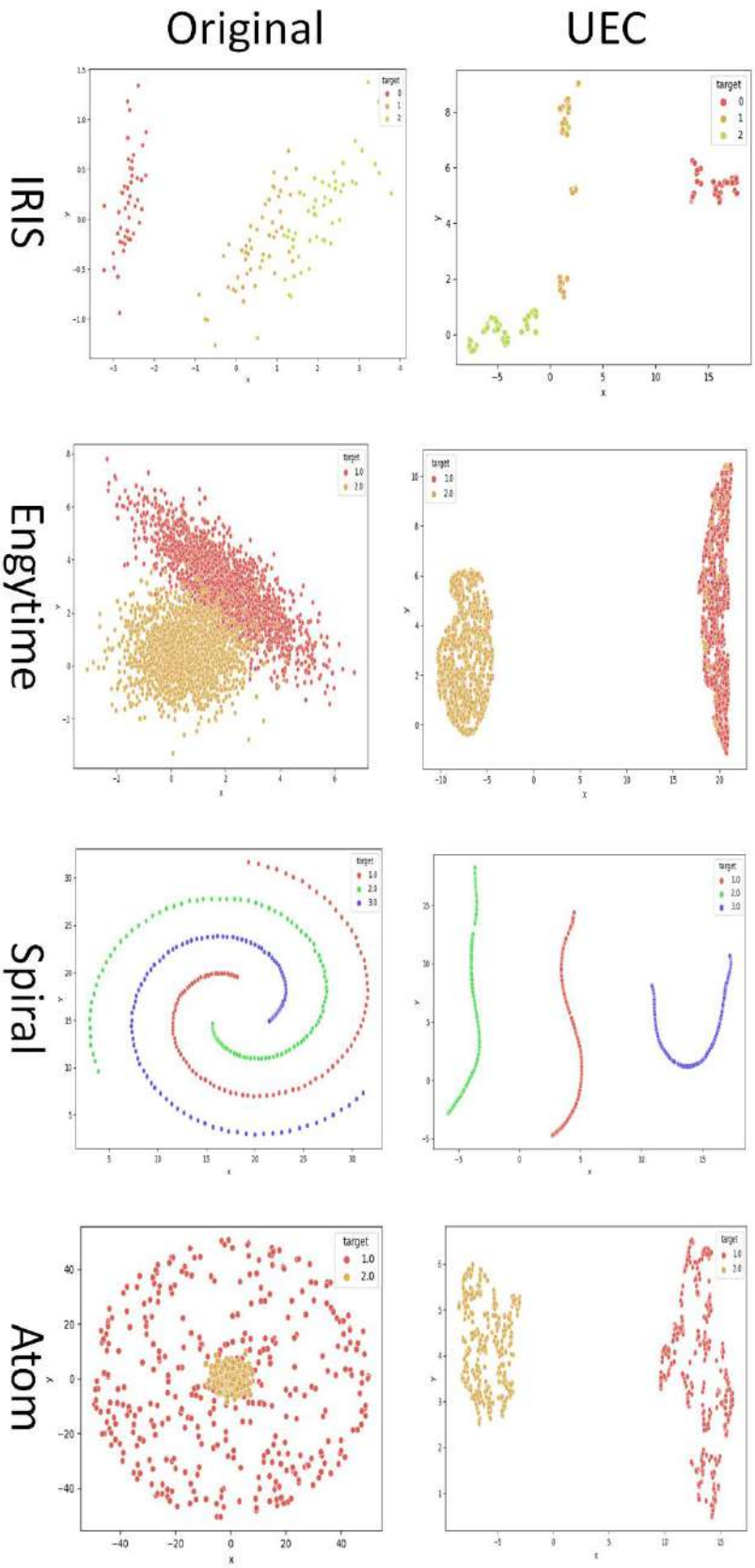

Figure 4.5 – visualization of challenging datasets: Original vs. The UEC visualization.

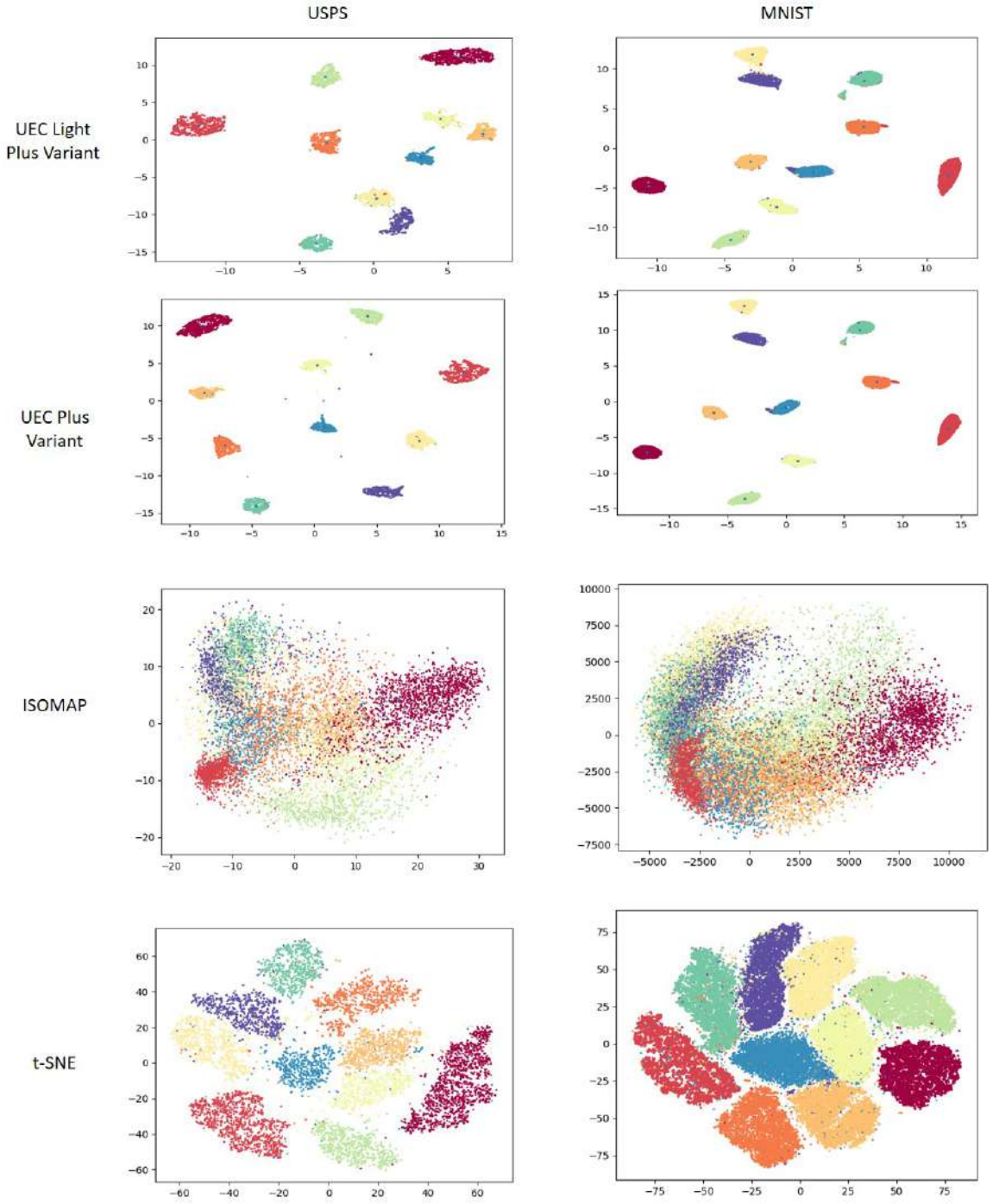

Figure 4.6 – The visualization of USPS and MNIST datasets using the UEC against ISOMAP and t-SNE.

4.6 Conclusion

In this chapter, We have presented the UEC technique, which jointly optimizes the representation and clustering of data. UEC is a manifold-based clustering algorithm that can preserve data's local and global structure and seeks to learn the manifold within the embedding space. It comes with three variants that resulted from the optimization process: *Comma variant, Plus variant,* and *Light plus variant.* The empirical results obtained through performance and sensitivity analysis have shown the high effectiveness of UEC across several large-scale benchmarks and against some baseline algorithms.

In the next chapter, we will talk about applying embedding and clustering techniques to solve problems related to the COVID-19 pandemic. Exactly, we will show how embedding and clustering techniques help in organizing and visualizing the COVID-19-related literature.

Chapter 5

A Machine Learning-Based Tool for Exploring COVID-19 Scientific Literature

5.1 Introduction

In the last few years, and since 2019, the world has witnessed the outbreak COVID-19 pandemic. This pandemic seriously threatened all humans, causing hundreds of millions of cases in which a few million have died [77, 78]. This has caused saturation in health systems even in the most developed countries, an unprecedented economic recession, the closure of borders between countries, the locking of stores, and the unavailability of many services people may need [79, 80].

Considerable efforts are made to counter the spread of this pandemic and limit its damage. Thousands of researchers from all over the world have actively participated in this effort. They tried, each in their discipline, to develop new vaccines, purpose drugs, produce tools for tracking contamination among the population, study the effects of the lockdown on the economy of countries and the psychology of individuals, etc.

The current work aims to use ML techniques to build an efficient tool for automatically organizing and visualizing COVID-19-related scientific documents. The developed tool is

handy for researchers and decision-makers since it allows them to:

— Navigate easily through the huge number of documents in the dataset,

— Easily find all the documents related to a given topic,

— Locating the most relevant documents to a given paper.

The rest of the paper is organized as follows: In Section 5.2, we will give details about our method, and in Section 5.3, we will report some experimental results. Then, The paper finishes with concluding remarks and perspectives for future research in Section 5.4.

5.2 Proposed Method

As we said above, our goal is to develop a tool for organizing and visualizing the scientific documents related to COVID-19 based on the recent Machine learning techniques. Ultimately, the documents are represented in a 2D graphical space where the user can navigate through the whole collection of documents. Related documents should be visualized close to each other [11]. In addition, each document (a point in our 2D space) is provided with its metadata: title, author names, abstract, and other valuable details.

In Fig. 5.1, we present the pipeline of our tool, which can be summarized in the following steps:

— First, documents are pre-processed to eliminate stop words and useless ones. In addition, papers written in languages other than English are ignored. Details are given in Section 5.2.1.

— Second, we do Feature Extraction using Term Frequency–inverse document frequency (Tf-idf) technique [81]. This is detailed in Section 5.2.2.

— Third, we perform dimensionality reduction using a deep Denoising Autoencoder [13], as explained in Section 5.2.3.

— Fourth, the autoencoder output is projected into a 2D space using UMAP embedding technique [12]. Then, we use the Agglomerative clustering algorithm [30] to cluster the output of UMAP. See Section 5.2.4.

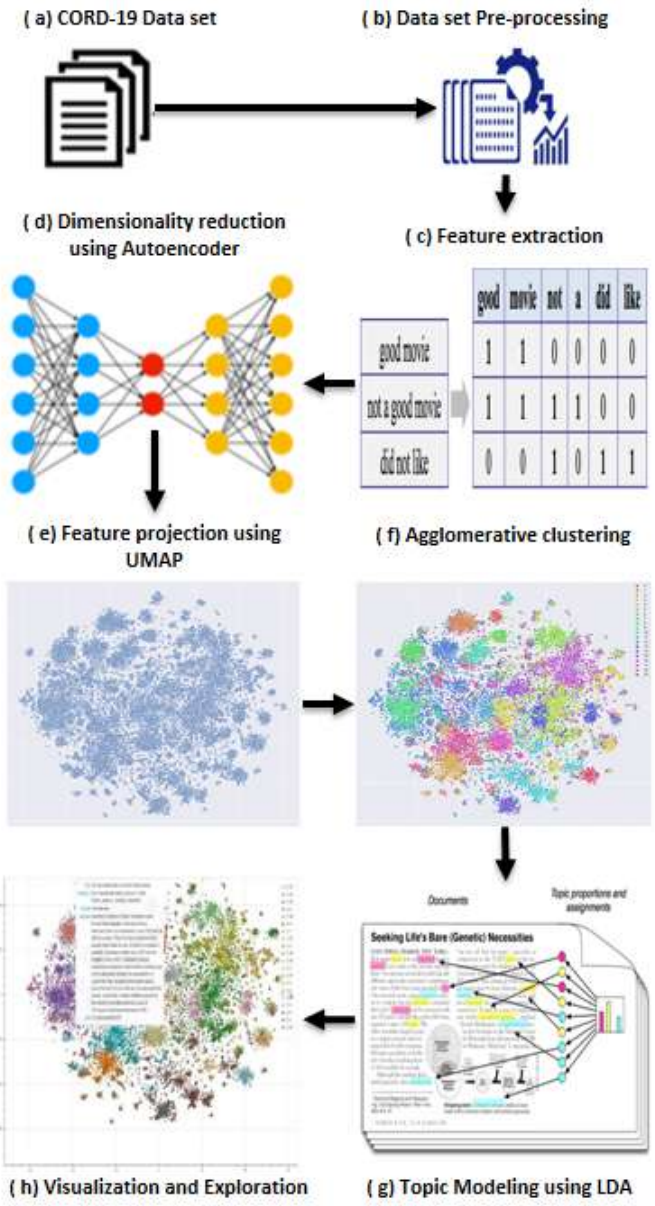

Figure 5.1 – The steps of our method: (a) CORD-19 data set (b) Pre-processing: We keep the documents written in English only, and then we remove the punctuation and stop-words. (c) Features are extracted using Tf-idf Vectorizer, (d) The dimensionality of the extracted features is reduced using a deep Denoising Autoencoder. (e) The reduced data are projected into a 2D space using UMAP (f) and then clustered using the Agglomerative algorithm. (g) We use LDA to perform topic modeling and extract the keywords that best represent each cluster. (h) Finally, the whole dataset is viewed by the user through an interface where he can navigate through topics and papers.

— Sixth, we perform Topic modeling using Latent Dirichlet Allocation LDA [82] as explained in Section 5.2.5. The goal of this step is to label each cluster automatically.

— Finally, the results are visualized to the user through our interface as depicted in Section 5.2.6.

In the following subsections, we detail each step.

5.2.1 Pre-processing

The pre-processing step is essential because it permits data filtering before processing. When adequately done, it improves any ML algorithm's performance. First, we load the dataset and extract the text of the papers, making it up. We then remove any document that is not in English, and from the retained documents, we remove any punctuation or stop-word. Such words are irrelevant in our case and constitute noise that may mislead the algorithm.

5.2.2 Feature Extraction

The documents are in text format and need to be converted to a numerical format that our algorithm can handle. This conversion is done through the well-known Term Frequency–inverse document frequency (Tf-idf) [81] document feature extraction technique. It transforms row text into fixed-size numerical feature vectors based on the frequency of the different words in each document.

5.2.3 Dimensionality reduction

The features obtained from the above step are of a very high dimension, which equals the number of possible words in the dataset. Since then, a dimensionality reduction has been mandatory to make them usable by embedding and clustering techniques. Denoising Autoencoder (DAE) [13] is a good dimensionality reduction technique that can extract the intrinsic structure of the dataset. We trained a DAE to be able to reconstruct the documents of our collection. DAE contains an encoder and decoder parts, and each layer in the encoder is represented as follows:

$$\tilde{x_i} \sim Dropout(x_i)$$

$$e_i = (f_{e_i}(W_{e_i}\tilde{x} + b_{e_i})$$

where x_i is the input of the $i-th$ encoding layer. Dropout(.) [83] is a regularization technique used to reduce the over-fitting in our autoencoder. f_{e_i} is the activation function of the current encoding layer, and $\theta_{e_i} = \{W_{e_i}, b_{e_i}\}$ are the parameters of the $i-th$ encoding layer.

The following equations represent the layers of the decoder part:

$$\tilde{e_i} \sim Dropout(e_i)$$

$$d_i = (f_{d_i}(W_{d_i}\tilde{e} + b_{d_i})$$

where e_i is the input of the $i-th$ decoding layer, f_{d_i} is the activation function of the current decoding layer, and $\theta_{d_i} = \{W_{d_i}, b_{d_i}\}$ are the parameters of the $i-th$ decoding layer. Rectified Linear Units (ReLUs) [84] are used as an activation function, except for the encoder's input layer and the decoder's output layer. We use the least-squares loss as an objective function:

$$\min \|x - e\|_2^2$$

5.2.4 Projecting and Clustering

Inspired from [49], we use the Uniform Manifold Approximation and Projection (UMAP) technique [12] to project the features extracted by the deep Autoencoder into a 2D embedding space. More details about UMAP are presented in section 3.2.

We follow the projection step with the clustering step using Agglomerative Hierarchical clustering [30], an effective clustering algorithm that doesn't require specifying the number of clusters apriori.

5.2.5 Topic Modeling

Assigning labels to each cluster generated by the previous step is essential. Indeed, clustering algorithms group similar data into clusters without providing labels or information about clusters. This would make our cluster dumb. We, therefore, resorted to topic modeling to automatically extract the keywords that represent each cluster accurately.

We used the Latent Dirichlet Allocation (LDA) topic modeling technique [82]. LDA is a popular method used in a variety of applications. It is a simple and effective method that can summarize the content of large datasets. In LDA, each document can be described by a distribution of topics, and a distribution of words can describe each. It is a generative statistical model that allows sets of words to be explained by a shared topic. LDA extracts each topic's keywords, which are later used as labels.

5.2.6 Visualization Tool

Our tool has a user-friendly interface that visualizes the whole dataset, as shown in Fig. 5.2. Different clusters are visualized in different colors. In addition, documents dealing with the same or similar subjects are viewed close to each other. This greatly helps the user go from a given document to a similar one. Finally, the user can visualize the metadata related to the documents he wants, including the document's title, authors, and abstract. See the floating message box in Fig. 5.2.

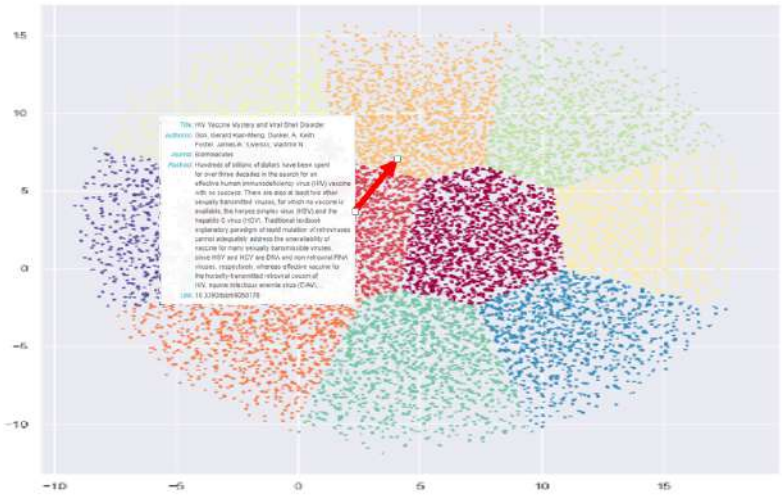

Figure 5.2 – Visualization of the literature related to COVID-19.

5.3 EXPERIMENTAL EVALUATION

5.3.1 Evaluation Protocol

We validated our method on the COVID-19 Open Research Dataset (CORD-19) [58], which is presented in Sec. 2.5. However, This dataset does not contain ground truth or any other mechanism that enables evaluating organization algorithms developed for it. Labeling the whole dataset with ground truth is a very tedious task and out of the scope of this work. Since then, we have adopted some evaluation methods that do not require labeling. We conducted 3 experiments:

— The first experiment will measure the homogeneity of our results compared to those of other algorithms and the fact that the documents belonging to a given cluster should talk about the same topic. We used the accuracy measure based on the Hungarian algorithm [59].

— The second experiment will assess the internal validity of our clustering algorithm and compare it with the same algorithms as the first experiment.

— The third experiment will show how we determined the optimal number of clusters.

5.3.2 First experiment: Homogeneity of our clusters

In the first experiment, we adopted the same protocol as [47], which can be summarized as follows:

1. We first applied our algorithm to organize the dataset into clusters and predict cluster labels.
2. We then partitioned each of the obtained clusters into two parts: train and test.
3. We finally trained three classifiers on the train part, then tested on the test ones, and then calculated their accuracy. The used classifiers are SGD [85], KNN [86], and SVC [87].

Their accuracy should allow for assessing the homogeneity of clusters, i.e., all the documents belonging to a given cluster should deal with the same topic. The rationale behind this experiment is as follows: if we train an excellent classifier on the –labeled– train part of any given cluster, then when tested on the test part of the same cluster, this

classifier must be able to predict the same label. Consequently, the higher the accuracy of the classifier, the more homogeneous our clusters are. The results of this experiment are shown in Fig. 5.3 where we compared our algorithm against a few other methods respectively: Deep Embedded Clustering DEC [7], Birch [88], Spectral Clustering [33], and the one adopted in COVID-19 LC [47]. This latter is similar to ours since it is dedicated to COVID-19-related literature.

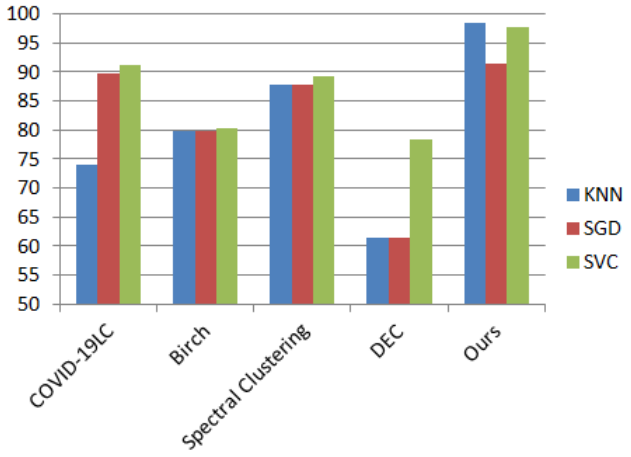

Figure 5.3 – Result of clustering algorithms in terms of accuracy.

From Fig. 5.3, we can observe that, when applied to our algorithm's results, the three classification algorithms gave the best accuracy compared to the other algorithms. This can be explained by the fact that other algorithms preserve only the global structure of the dataset. On the other hand, our choice to apply UMAP [12] before clustering helped preserve both the dataset's local and global structure. In other words, UMAP finds the similarities between documents in the original space and then tries to preserve them in the embedding space. This allowed us to produce a clustering-friendly space and permitted our algorithm to produce coherent clusters regarding topics.

The same figure also shows that the Spectral clustering algorithm performed relatively well. This can be attributed to the fact that this algorithm preserves local similarity between documents, like UMAP.

5.3.3 Second experiment: Internal Clustering performance of our algorithm

In this experiment, we aim to compare our algorithm with the same algorithms of the first experiment regarding internal clustering performance without needing labels. The results are given in Table 5.1, where three measures are Davies-Bouldin (DB) [63], Calinski-Harabaz (CK) [64], and Silhouette Coefficient (SC) [65]. More details about these metrics are presented in section 2.5.2

Table 5.1 – Comparison with different algorithms in terms of Davies-Bouldin (DB), Calinski-Harabasz (CH), and Silhouette Coefficient (SC).

Clustering Methods	Clustering measures		
	DB	***CH***	***SC***
COVID-19 LC	5.83	628.48	0.016
Birch	6.50	155.06	0.007
Spectral clustering	5.75	313.20	0.014
DEC	25.99	315.59	0.009
Ours	**1.22**	**10644.78**	**0.359**

According to Table 5.1, we first notice that our algorithm yields significantly better results than all the other ones in terms of all indices. Far behind our algorithm, two algorithms snatch the second place: Spectral clustering [33], which is second in terms of DB index and third in terms of SC, and COVID-19 LC [47], which is second in terms of CH and SC and third in terms of DB. On the other hand, the less efficient algorithms are Birch [88] and DEC [7]. This experiment assesses the quality of our algorithm's clustering, ensuring a good dataset organization.

5.3.4 ELBOW method to find the correct number of clusters

The correct number of clusters the CORD-19 dataset comprises is unknown. Since then, it has been necessary to use an automatic method to discover this number, which we will call K. We used the ELBOW method [81]. We run our algorithm several times, changing the value of K each time and measuring the distortion. This latter is the average of the squared distances from the cluster centers. Fig. 5.4 plots distortion against K. At the beginning, the distortion decreases quickly with the increase of K, and then there

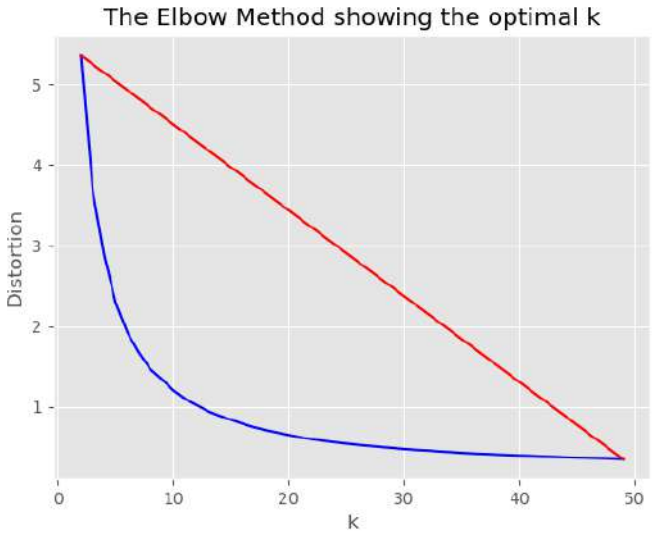

Figure 5.4 – ELBOW method to find the best value of K: the number of clusters.

is no noticeable decrease. The better value of K is situated in the zone, looking like an elbow in this curve. In our case, we observe that this value comprises between 5 and 15. This is why we set $K = 10$ in the first two experiments.

5.4 Conclusion

In this chapter, we presented a new method for the automatic organization and bibliometric analysis of the many scientific papers related to COVID-19 that have been published recently. The proposed method first represents those documents numerically, then clusters them, and finally discovers the topic that each cluster deals with. Our method uses some of the most recent and advanced machine learning techniques, such as UMAP embedding and Deep autoencoders. The tool we developed helps researchers navigate all related publications easily and discover all notably similar studies. Besides, cluster mapping can be helpful for data analysis. The conducted experiments proved that our method achieved outstanding results. The next chapter presents an entirely novel method for the organization and the analysis of COVID-19-related documents.

Chapter 6

A novel two-fold loss function for data clustering and reconstruction: application to document analysis

6.1 Introduction

In the previous chapter, we discussed the problem posed by the emergence of the COVID-19 pandemic. We also presented a solution, which is a combination of algorithms for dimensionality reduction, embedding, clustering, and topic modeling. However, much effort remains to be made to improve the effectiveness of the organization of documents related to COVID-19. To this end, we present our novel deep dimensionality reduction and data clustering architecture in this chapter.

The rest of this chapter is organized as follows: More details about our method are in Sec. 6.2. Section 6.3 reports some experimental results. Then, we conclude the paper in Sec. 6.4.

6.2 Proposed Method

This section is devoted to presenting the proposed method in detail. Indeed, the main target of this study is to develop a clustering method to organize the scientific documents related to COVID-19 based on deep learning. To do so, we propose a novel deep architecture that simultaneously carries out dimensionality reduction and data clustering (DJRC). Figure 6.1 presents the general flowchart of the proposed approach. From Figure 6.1, we can notice that the first step of our method is to pre-process the input documents by eliminating punctuation, stop words, orthographic and spelling errors, symbols, etc. We consider an autoencoder-based architecture in which three components, two different encoders and one decoder, are jointly trained. We refer to the two encoders as clean and noisy because the former is fed with pre-processed documents, whereas the last one is fed with raw documents. To improve the model robustness, we consider training the noisy encoder with the weights of the clean encoder. To further improve the model robustness, inspired by the denoising auto-encoder (DAE) principle, we consider a two-fold loss (i.e., objective function), where the first fold of the loss is generated by considering the latent space of the noisy encoder and the output of the decoder. The second fold of the loss is designed for predicting the cluster assignments. To do so, we associate each clean and noisy encoder with two matrices, aiming to converge the matrices to each other by iteratively updating them during training. This double-side convergence can be achieved by considering the cross-entropy function instead of the conventional KL divergence. Indeed, this two-fold loss (i.e., objective function) allows us to reduce the dimensionality and cluster the data simultaneously. The cornerstone of the proposed method lies in this simultaneity, which allows our model to train efficiently and overcome the issue of latent space distortion.

After the training process had finished, and unlike the existing works, which fed the entire dataset documents to the topic modeling model, we fed the LDA with the data clusters, which are supposed to be consistent and group the documents covering the same topic, as shown in Figure 6.2. Doing so can significantly improve the model efficiency and reduce the response time, which is crucial for such models and for fighting COVID-19 in

general.

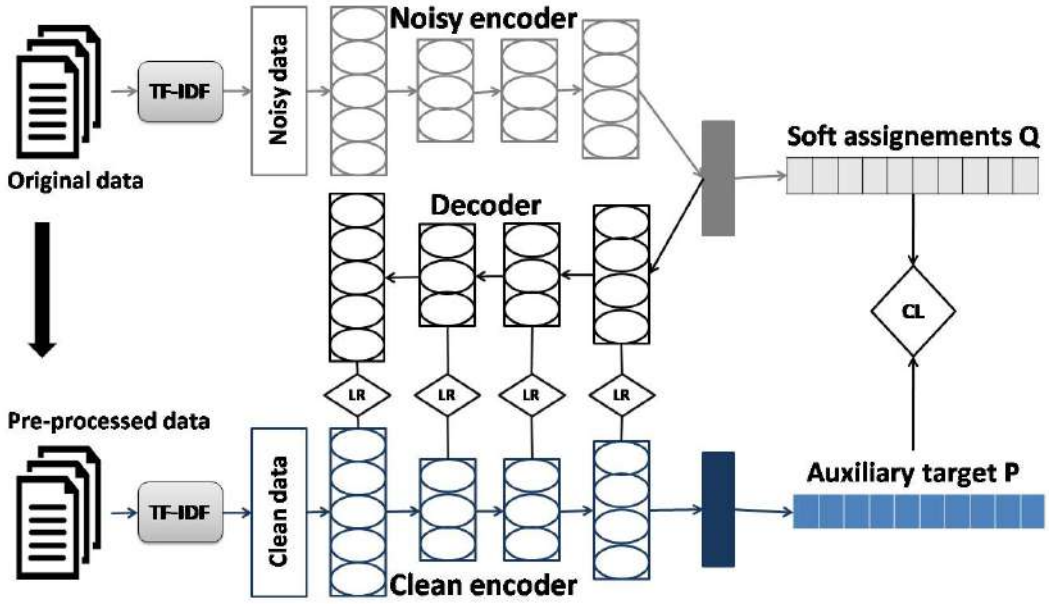

Figure 6.1 – DJRC's architecture. The clean and noisy data are passed to dimensionality reduction and clustering algorithms to embed and cluster the dataset.

6.2.1 Dataset pre-processing

In our work, we use the COVID-19 Open Research Dataset (CORD-19) [58], which is a freely and publicly available dataset on the Kaggle website. It contains over 1,000,000 scholarly articles, including over 400,000 full-text scientific papers concerning COVID-19, SARS-CoV2, and other related coronaviruses. In the natural language processing field, the pre-processing step (such as removing the punctuation, stop-word, etc.) is essential to filter and clean the data from inaccuracies, errors, or conflicting information to improve the performance of the proposed model [89]. Furthermore, we used the well-known Term Frequency–inverse document frequency (Tf-IDF) algorithm [81] to convert the documents from the text format to the feature vectors that can be processed by learning models later on.

TF-IDF vectorization involves calculating the TF-IDF score for every word in the corpus relative to that document and then putting that information into a vector. Thus each document in the corpus would have its vector, and the vector would have a TF-IDF score for every single word in the entire collection of documents. Then the similarity of the documents can be computed using cosine similarity between the vectors of those

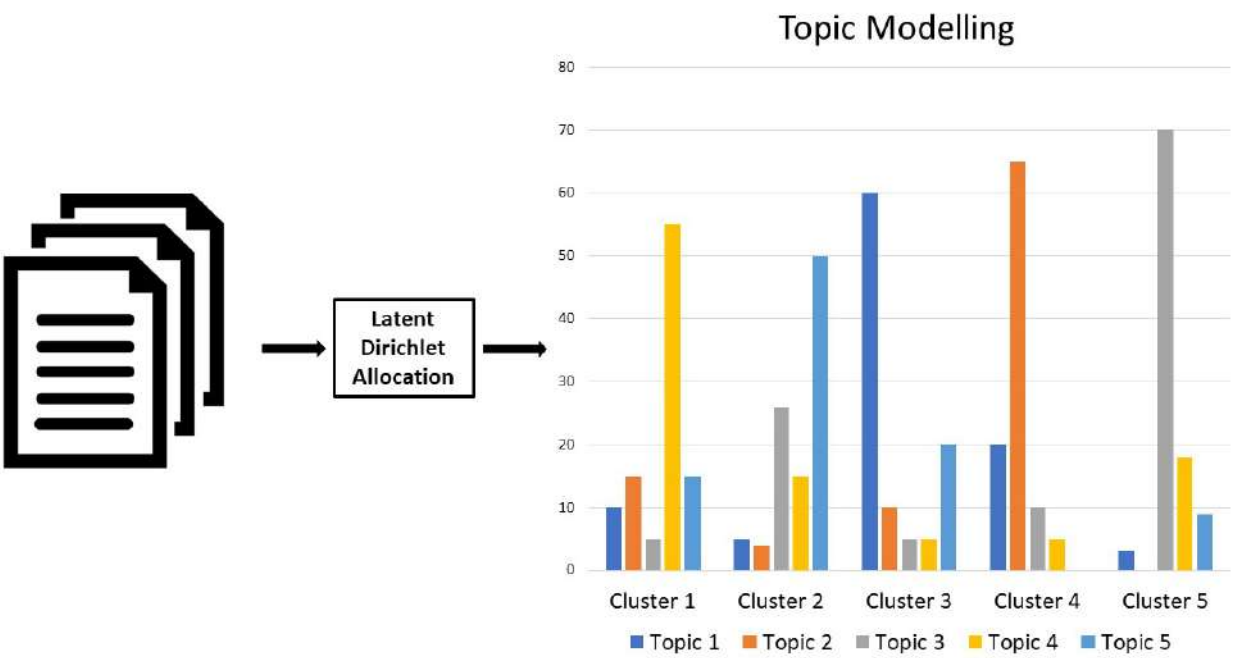

Figure 6.2 – Topic Modeling step. We use LDA to model the topic and extract the keywords that best represent each cluster topic.

documents.

6.2.2 Dimensionality Reduction and Cluster assginments

6.2.2.1 The proposed architecture

In our model, Deep Joint dimensionality reduction and clustering (DJRC) is based on a Denoising Autoencoder (DAE) with three components. DAE is a good dimensionality reduction technique that can extract the dataset's intrinsic structure [13]. Unlike the standard learning approach for denoising autoencoders, we built an autoencoder similar to the one reported in [8]. Figure 6.1 (a) represents the structure of the DAE, which contains three parts: a clean encoder, a noisy encoder, and a decoder part. The clean encoder is used to compute the more accurate target variables, while the noisy encoder is trained to achieve noise-invariant predictions. The clean and noisy encoders are trained together with the decoder, where the clean encoder shares its weights with the noisy encoder. In the following, we give more details on these deep architectures.

Noisy encoder This component is trained using noisy input data. Noisy data refers to text containing errors, inconsistencies, irrelevant information, punctuation, stop-word,

numbers, special characters, capitalization, etc. The aim of training an autoencoder using noisy data is to make it more robust to variations in the input data and to increase its ability to generalize to new, unseen data. When the autoencoder is trained on noisy data, it must learn to reconstruct the original, clean data from a corrupted version. This helps the model to become more robust to noise and to ignore irrelevant details in the input data. The following equation indicates the output of each layer in the noisy encoder:

$$\tilde{z}^l = f_e^l(W_e^l \tilde{z}^{(l-1)} + b_e^l) \tag{6.1}$$

$$\tilde{z}_l \sim Dropout(\tilde{z}_l) \tag{6.2}$$

where $\tilde{z}^l$ is the noisy input of the l-th encoding layer. Dropout(.) is a regularization technique that we used to reduce the over-fitting in our autoencoder. f_e^l is the activation function (Rectified Linear Unit (RELU) in our case) of the current encoding layer, and $\theta_e^l = \{W_e^l, b_e^l\}$ are the parameters of the l-th encoding layer, where W stands for the weights, and b represents the bias. The structure of the noisy encoder is $[D-500-500-2000-d]$, where D is the dimension of the input data, and d is the dimension of the latent space.

The noisy encoder is associated with a Soft assignment matrix denoted by Q. The Soft assignment matrix was computed using an equation similar to the one used in [12]. This equation measures the similarity between embedded point $\tilde{z}$ and centroid μ_k:

$$q_{ik} = (1 + d_{ik}^2)^{-1} \tag{6.3}$$

where d_{ik}^2 is the squared distance between the data points $\tilde{z}$ and the centroids μ_k, $d_{ik} = (\tilde{z} - \mu_k)$. $\tilde{z} = f(x_i) \in Z$ corresponds to the input data $x_i \in X$ after embedding.

The complexity of the noisy encoder is $O(N * W * e + Q)$, where N is the number of samples, W is the weights of the noisy encoder, and e is the number of epochs.

Clean encoder The clean encoder is trained using data cleaned from noise or corruption that misleads the learning process. The aim of training an autoencoder using clean data

is to learn a representation of the input data that is as accurate as possible. When the autoencoder is trained on clean data, it can learn the underlying structure of the data, including its important features and patterns. This allows the model to capture the input data's essence and accurately reconstruct the original data from its internal representation. The features of the clean encoder are used in the reconstruction loss function, which is inferred using the following equation.

$$z^l = f_e^l(W_e^l z^{(l-1)} + b_e^l) \tag{6.4}$$

where z^l is the input of the l-th encoding layer. f_e^l is the activation function of the current encoding layer, and $\theta_e^l = \{W_e^l, b_e^l\}$ are the parameters of the l-th encoding layer. The structure of the clean encoder is similar to the noisy encoder $[D-500-500-2000-d]$.

This component (i.e., the clean encoder) is associated with a matrix (denoted by P) which is iteratively updated during training epochs to achieve the cluster assignments. P represents the probabilistic point-to-cluster assignments using the obtained Soft assignment Q. Therefore, P is the auxiliary target distribution proposed to improve feature representation and clustering assignment.

$$p_{ik} = \frac{q_{ik}/\sum_l q_{lk}}{\sum_m (q_{lm}/\sum_l q_{lm})} \tag{6.5}$$

m is the number of clusters, and l is the number of data points in the corresponding cluster.

The complexity of the clean encoder is $O(N * W * e + P)$, where N is the number of samples, W is the weights of the clean encoder, and e is the number of epochs.

Decoder The decoder is a commune part between the noisy and the clean encoders. By training on clean and noisy data, the decoder can learn a more robust representation of the data and generalize better to new, unseen data. The following equation represents

the layers of the decoder part:

$$\hat{z}^l = f_d^l(W_d^l \hat{z} + b_d^l) \tag{6.6}$$

Where z_l is the input of the l-th decoding layer, f_{d_l} is the activation function of the current decoding layer, and $\theta_d^l = \{W_d^l, b_d^l\}$ are the parameters of the i-th decoding layer. The structure of the decoder is $[d - 2000 - 500 - 500 - D]$, where D is the dimension of the input data, and d is the dimension of the latent space. The complexity of the decoder is $O(N * W * e)$, where N is the number of samples, W is the weights of the decoder, and e is the number of epochs.

6.2.2.2 Joint optimization using the Two-fold objective function

The bottleneck of the proposed algorithm is to provide a joint learning framework that optimizes the binary cross-entropy and autoencoder parameters. This is achieved by using a two-fold function, which is given by

$$min \sum_i \sum_l \|z_i^l - \hat{z}_i^l\|_2^2 + \sum_i \sum_k p_{ik} \log \frac{p_{ik}}{q_{ik}} - (1 - p_{ik}) \log \frac{1 - p_{ik}}{1 - q_{ik}} \tag{6.7}$$

Training the autoencoder using reconstruction and clustering losses simultaneously ensures the achievement of both objectives appropriately. The reconstruction loss helps to ensure that the autoencoder can effectively capture the underlying structure of the data and reduce it to a lower-dimensional representation. The clustering loss helps to force the encoder to produce a compact and well-separated representation of the data in the latent space, making it easier to perform the clustering task. The first term in Eq. (6.7) is the Reconstruction Loss (RL):

$$RL = \sum_i \sum_l \|z_i^l - \hat{z}_i^l\|_2^2 \tag{6.8}$$

It is worth mentioning that the reconstruction loss is obtained by considering the weights from the clean encoder and the features (i.e., of the latent space) from the noisy encoder.

Combining the noisy encoder features and the clean encoder weights optimizes the reconstruction loss in a way that ensures the balance of the trade-off between better handling of the input data variations and the accuracy of the model.

The second term is the Clustering Loss (CL), computed between the Soft assignment matrix Q and the auxiliary target matrix P. The CL is optimized until the Soft assignment matrix approximates the auxiliary target matrix. The CL term is defined as

$$CL(P||Q) = \sum_i \sum_k p_{ik} \log \frac{p_{ik}}{q_{ik}} - (1-p_{ik}) \log \frac{1-p_{ik}}{1-q_{ik}} \tag{6.9}$$

We use the binary cross-entropy function as a clustering loss to perform double-side convergence, i.e., P converges to Q, and Q converges to P, which is inspired by UMAP. The Clustering Loss (CL) computes the total entropy between the quantity matrices Q and P.

The aim is to find the derivative of CL function w.r.t z_i and μ_k. First, CL was noted as a function having $z_1, z_2, \ldots, z_n, \mu_1, \mu_2, \ldots, \mu_C$ as vector variables. Then, the data point coordinates z_i are updated using the following iterative relations:

$$z_i^{(t+1)} = z_i^{(t)} - \eta \Delta_{z_i} CL(z_1^{(t)}, z_2^{(t)}, \ldots, z_i^{(t)}, \ldots, z_N^{(t)}, \mu_1^{(t)}, \mu_2^{(t)}, \ldots, \mu_k^{(t)}, \ldots, \mu_C^{(t)}), \tag{6.10}$$

for $i = 1, ..., N$ and t is the time step. Where η is the learning rate, $\Delta_{z_i} CL$ is the gradient of CL loss function with respect to the components of the vector z_i i.e., if $z_i = (z_{i1}, z_{i2}, \ldots, z_{id})^T$ then $\Delta_{z_i} CL = (\frac{\partial CL}{\partial z_{i1}}, \frac{\partial CL}{\partial z_{i2}}, \ldots, \frac{\partial CL}{\partial z_{id}})^T$ where $d = 1, ..., d$ is the dimension of data space and the subscript T designate the transpose of the vector. The partial derivative of the CL loss function w.r.t each component d of the vector z_i is:

$$\frac{\delta CL}{\delta z_i} = \sum_k \left[\frac{2p_{ik}}{1+d_{ik}^2} - \frac{2(1-p_{ik})}{d_{ik}^2(1+d_{ik}^2)} \right] (z_i - \mu_k) \tag{6.11}$$

Similarly, the coordinates of the centers of the clusters μ_k are updated using the

following iterative relations:

$$\mu_k^{(t+1)} = \mu_k^{(t)} - \eta \Delta_{\mu_k} CL(z_1^{(t)}, z_2^{(t)}, \ldots, z_i^{(t)}, \ldots, z_N^{(t)}, \mu_1^{(t)}, \mu_2^{(t)}, \ldots, \mu_k^{(t)}, \ldots, \mu_C^{(t)}), \quad (6.12)$$

for $k = 1, ..., C$ and t is the time step. Where η is the learning rate, $\Delta_{\mu_k} CL$ is the gradient of CL loss function with respect to the components of the vector μ_k i.e., if $\mu_k = (\mu_{k1}, \mu_{k2}, \ldots, \mu_{kd})^T$ then $\Delta_{\mu_k} CL = (\frac{\partial CL}{\partial \mu_{k1}}, \frac{\partial CL}{\partial \mu_{k2}}, \ldots, \frac{\partial CL}{\partial \mu_{kd}})^T$. The partial derivative of the CL loss function w.r.t each component d of the vector μ_k is:

$$\frac{\delta CL}{\delta \mu_k} = \sum_k \left[\frac{-2p_{ik}}{1 + d_{ik}^2} + \frac{2(1 - p_{ik})}{d_{ik}^2 (1 + d_{ik}^2)} \right] (z_i - \mu_k) \quad (6.13)$$

Stochastic Gradient Descent (SGD) was used to optimize the cluster centers μ_k and the parameters of the DAE jointly. For each iteration, the gradients $\frac{\delta CL}{\delta z_i}$ are passed down to the deep autoencoder and update mapping parameters using back-propagation for computing z_i.

6.2.3 Topic Modeling

In order to find the topics contained in the extracted clusters through our model to interpret and understand the data and gain insights into the reasons behind the model's decisions, we resorted to the topic modeling step. We used Latent Dirichlet Allocation (LDA) as a topic modeling technique. Unlike the previous studies, and to improve the computational efficiency of the proposed approach, we opt for using the clusters produced by the previous step (i.e., dimensionality reduction and clustering assignment). These clusters are fed to the LDA instead of the whole dataset. LDA [28] is a widely used method in various applications. In LDA, each document is described by a distribution of topics, and the distribution of words can describe each topic, as shown by Figure 6.3.

Many reasons make LDA a good choice as a topic modeling technique. LDA is flexible, effective, and scalable to handle large datasets such as CORD-19. LDA is a generative model that explicitly models the generation of documents based on a set of latent topics. This makes it easy to interpret the model's results and understand the relationships

between topics and documents. LDA is an unsupervised learning technique that does not require labeled data, which makes it useful in our case.

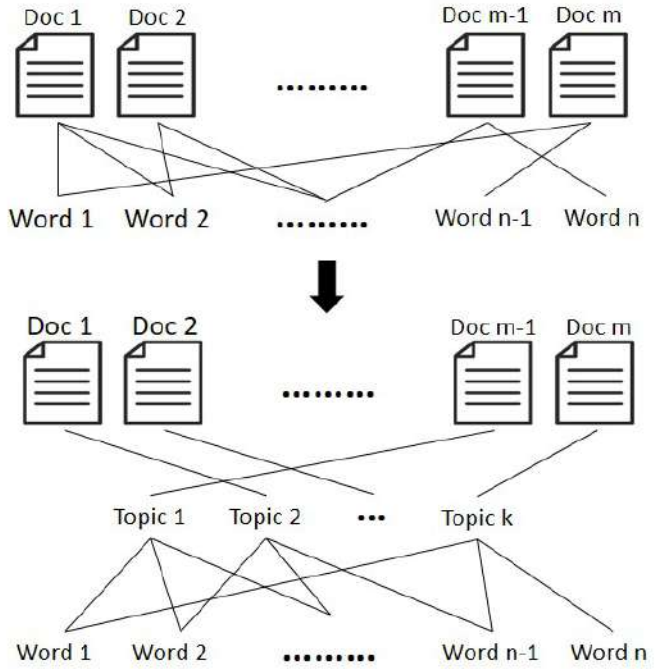

Figure 6.3 – Latent Dirichlet Allocation description.

6.3 Experimental results

In this section, we report our experimental findings. Specifically, we carry out several experiments to measure the performance of our proposed architecture. We divided these experiments into two studies: a comparative study and a case study, as follows

— In the Comparative study, we compare the performance of our model against several baseline models on benchmark datasets.

— In the case study, we study the performance of our model on an important topic: the documents related to COVID-19.

Let us first represent the evaluation metrics, the dataset we considered in these experiments, and the implementation details.

6.3.1 Experimental setup

We conduct experiments on four benchmark datasets USPS [54], MNIST-FULL [55], MNIST-Test, The COVID-19 Open Research Dataset Challenge (CORD-19) [58]. As

evaluation metrics, we used the ACCuracy (ACC) [59] and Normalized Mutual Information (NMI) in the comparative study, and We used Davis-Bouldin (DB)[62], Silhouette coefficient (S) [61] in the case study. In addition, we used Coherence Score C_v to evaluate the performance of LDA.

Implementation details

For feature extraction, our DAE is trained for 200 epochs, where the size of latent space equals the number of clusters, which is empirically determined in the first experiment. The output and the latent layer, all internal layers, are activated by ReLU. Finally, the autoencoder is trained using the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 1.0, gradually decreasing. In addition, we set the momentum to 0.9.

6.3.2 Comparative Study

6.3.2.1 Baseline Methods

The proposed DJRC is compared with a set of deep and manifold-based clustering methods including state-of-the-art deep clustering methods: k-means [6], deep embedded clustering (DEC) [7], Deep Embedded Regularised Clustering (DEPICT) [8], Deep Adaptive Clustering (DAC) [9], Deep Embedded Dimensionality Reduction Clustering (DERC) [10]. For these methods, the performance results are taken from the original publications.

6.3.2.2 Experiment results

Tables 6.1 and 6.2 outline the performance in terms of accuracy and NMI, respectively, where the top three accuracy scores are highlighted. Table 6.1 shows that both variants of DJRC are competitive with other algorithms across all benchmarks. It outperforms the mentioned deep clustering methods. In addition, DEPICT and DAC techniques are designed for image datasets, so these techniques cannot be performed on other datasets, such as document datasets. In contrast, DIRC can be applied to any dataset.

6.3.3 Case study using CORD-19 dataset

To assess the performance of the proposed method, we conduct experiments on the public CORD-19 dataset [58], described in Sec. 2.5.1. This dataset is made up of scientific

Table 6.1 – DJRC vs. other baselines: accuracy scores.

Models	Dataset		
	USPS	MNIST-FULL	MNIST-Test
k-means	0.668	0.572	0.570
DEC	0.619	0.843	0.841
DEPICT	0.964	0.965	0.963
DAC	0.972	0.978	0.975
DERC	0.977	0.975	0.976
DJRC	**0.979**	**0.981**	**0.979**

Table 6.2 – DJRC vs. other baselines: NMI scores.

Models	Dataset		
	USPS	MNIST-FULL	MNIST-Test
k-means	0.450	0.499	0.498
DEC	0.586	0.816	0.814
DEPICT	0.927	0.917	0.915
DAC	0.928	0.935	0.934
DERC	0.942	0.927	0.924
DJRC	**0.945**	**0.938**	**0.936**

documents that are concerned with discussing COVID-19 from different aspects. Note that the databases from which these documents are taken are in different languages, including English and other languages, as shown by Table 6.3. This Table also mentions the number of documents for each language. As a preliminary step, we detect the language of each document, and then we limit our attention to English documents. Figure 6.4 represents the number of words in different papers. As can be seen from this figure, most papers are about 5000 words in length. The existence of outliers causes the long tail in Fig. 6.4. Approximately 98% of the papers are under 20,000 words in length.

language	af	ca	cy	de	en	es	fr	it	nl	pl	pt	zh-cn
Number of papers	2	5	7	110	57602	290	336	18	40	3	15	3

Table 6.3 – The number of papers in each language. af: Afrikaans, ca: Catalan, cy: Welsh, de: German, en: English, es: Spanish, fr: French, it: Italian, nl: Dutch, pl: Polish, pt: Portuguese, zh-ch: Simplified Chinese.

6.3.3.1 Suitable number of clusters

It is very challenging to precisely determine the number of clusters in the CORD-19 dataset. This parameter is quite crucial, as the outcomes of the proposed method heavily depend on this parameter. In the case of an unknown number of clusters, the first step is

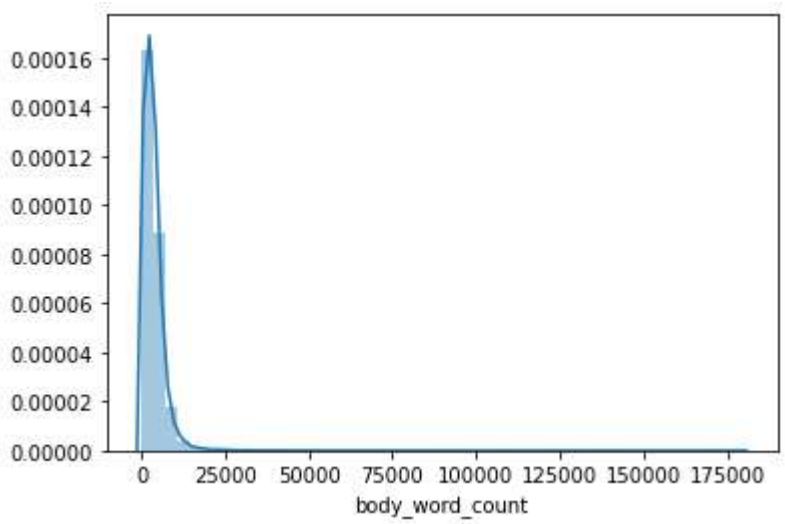

Figure 6.4 – Count of words in papers.

to specify a range of candidates (k) from which the exact number of clusters is picked out. This work sets this range to $[2, 50]$. In this experiment, we evaluate the performance of our algorithm for dimensionality reduction and clustering in terms of DB and SC measures when varying the number of clusters.

Figure 6.5 depicts the performance of the proposed method, in terms of DB and SC metrics, when varying the number of clusters. As shown in Fig. 6.5, the proposed method performs best for k between 5 and 10. The goal of clustering is to make the distance between two points belonging to two different clusters far apart (Inter-cluster distance) and to make points belonging to the same cluster close as possible (Intra-cluster distance). This is precisely what the Internal validation measures (i.e., DB and SC measures) are targeting to do based on the following two criteria: Separation and Compactness. The Separation measures how distinct or well-separated a cluster is from other clusters. The Compactness measures how closely related the objects in a cluster are. Consequently, the optimal number of clusters is the one for which the proposed algorithm performs best in DB and SC measures.

6.3.3.2 Evaluating the performance of clustering algorithm

In this experiment, we compare our method with the following methods:

— Deep Embedded Clustering DEC [7]

— Birch [88]

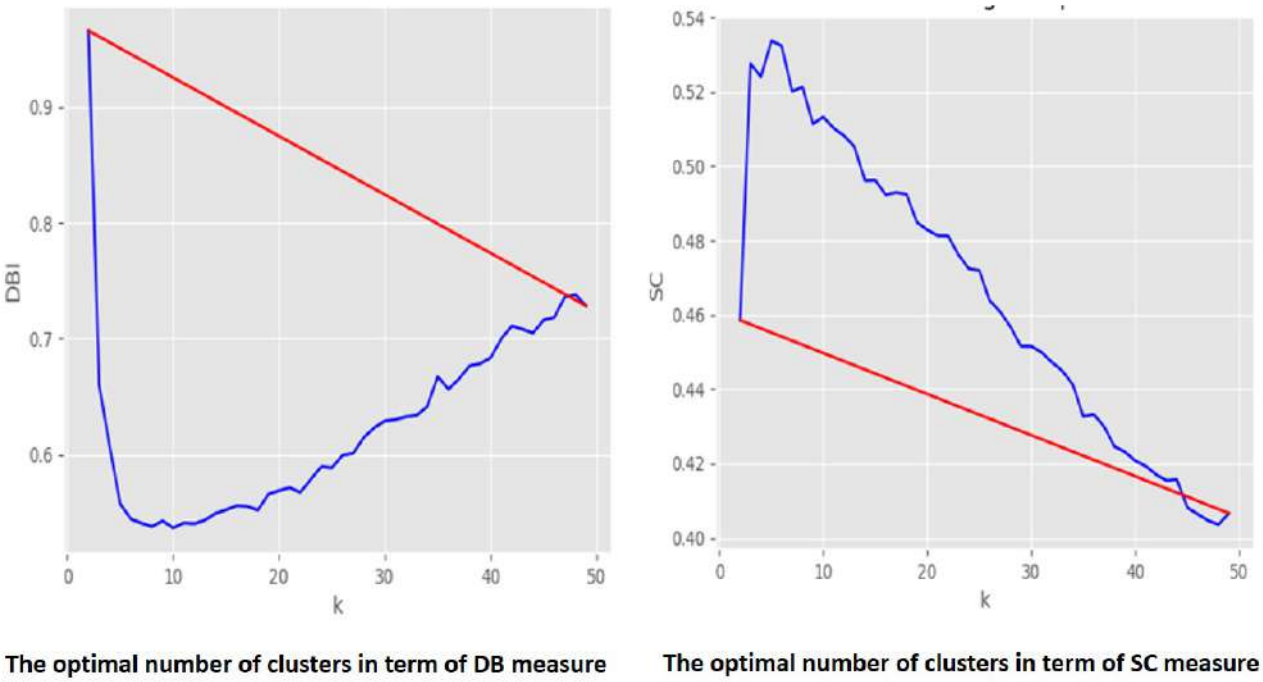

Figure 6.5 – Evaluating the performance of the proposed algorithm in terms of DB and SC when varying the number of clusters.

— Spectral Clustering [33]

— COVID-19 LC [47]

— and our previews contribution (presented in chapter 5) we called DAE+UMAP+AGG

Table 6.4 – Comparison with different algorithms in terms of Davies-Bouldin (DB) and Silhouette Coefficient (SC).

Clustering Methods	Clustering measures	
	DB	***SC***
COVID-19 LC [47]	5.83	0.016
Birch [88]	6.50	0.007
Spectral clustering [33]	5.75	0.014
DEC [7]	25.99	0.009
DAE+UMAP+AGG	1.22	0.359
Ours	**0.637**	**0.520**

Our evaluation aims to compare our model to the different approaches in clustering effectiveness using the DB and SC metrics. The comparison results are given in Tab. 6.4. We can notice through this table that our algorithm is significantly better than all the other algorithms in terms of all indices. These results assess the quality of our algorithm's clustering, which ensures outstanding dataset organization. We can observe

that DAE+UMAP+AGG came in second compared to the other methods regarding all indices. In DAE+UMAP+AGG, the combination of deep and manifold embedding techniques improves the performance of agglomerative clustering. In DAE+UMAP+AGG, and compared to the other models, the combination of deep and manifold embedding techniques has improved the performance of agglomerative clustering. Although DEC is a deep clustering method that is designed to deal with high-dimensional and non-linear data, we notice that the classical clustering approaches (i.e., Spectral clustering [33], COVID-19 LC [47] which use k-means as a clustering algorithm, and Birch [88]) generally outperform the DEC [7].

The learning process of the proposed deep learning model is based on two kinds of losses, namely reconstruction and clustering. We can understand how our deep learning algorithm represents the CORD-19 dataset through loss functions. If the predicted output of our model deviates too much from the actual data or output, the loss function will produce a high error value. Figure 6.6 shows the behavior of the two losses. This figure shows that the two losses softly decreased toward zero. This can be considered a good indication of the performance of our model.

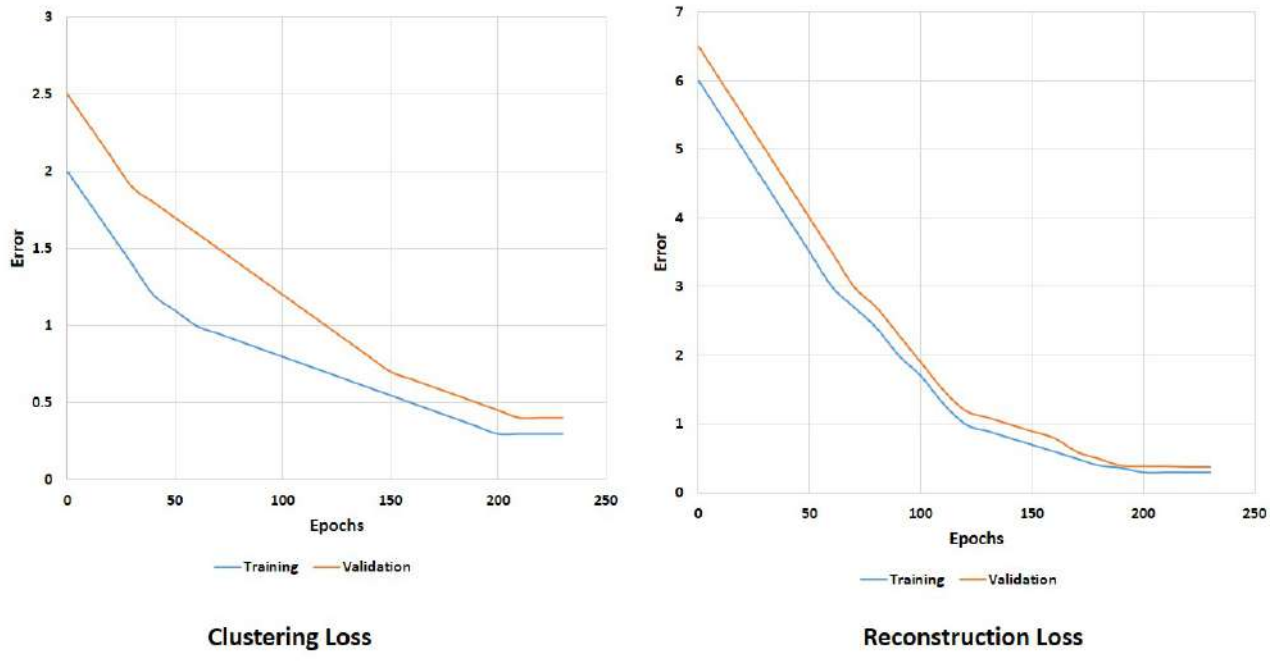

Figure 6.6 – The error of the clustering and reconstruction loss per epochs.

6.3.3.3 Model Interpretability

An important point we should study is understanding the model's performance and interpreting its results. In this part, we use each cluster's feature importance analysis technique. We used a model-agnostic approach called the Unsupervised to Supervised technique [90]. This technique converts the unsupervised clustering problem into a One-vs-All supervised classification problem using an interpretable classifier such as a tree-based model. The steps to do this are as follows:

— Change the cluster labels into One-vs-All binary labels for each

— Train a classifier to discriminate between each cluster and all other clusters

— Extract the feature importance from the model

After converting the problem into a binary classification problem, we chose a Random Forest Classifier for the next step, which is the importance of getting the features with the most discriminatory power between all clusters and the targeted cluster. Figures 6.7 and 6.8 present the achieved results.

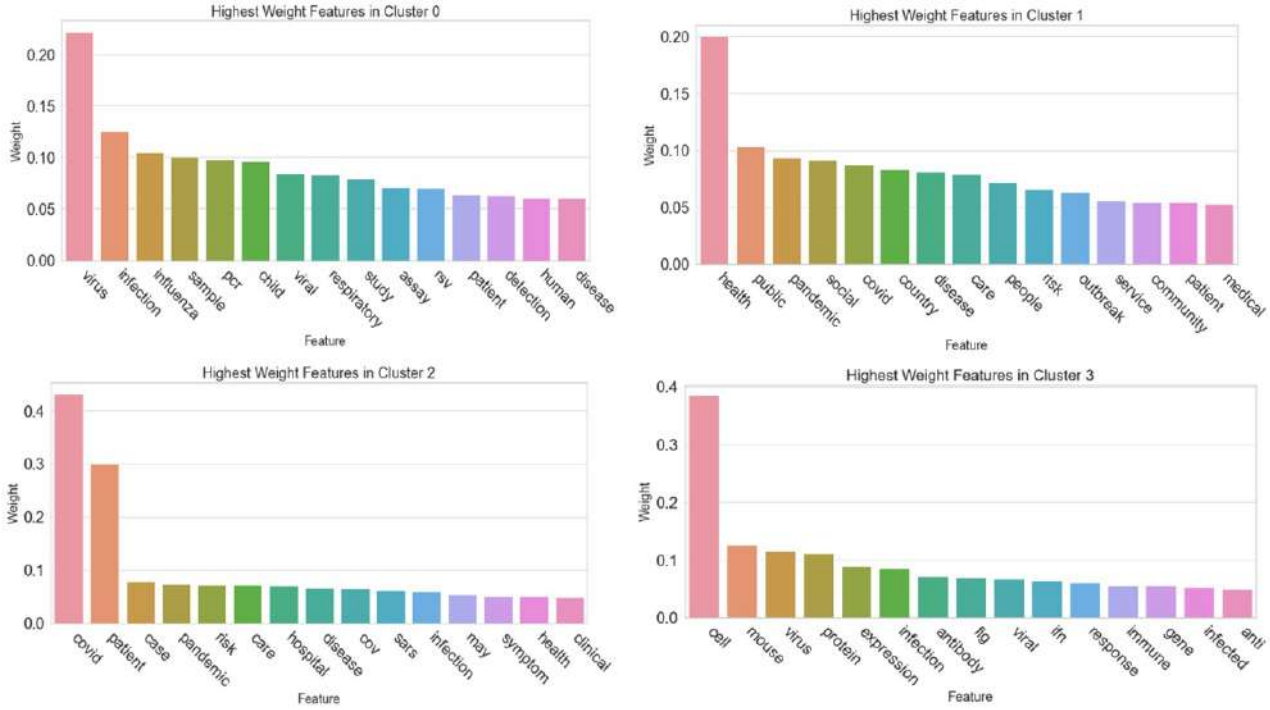

Figure 6.7 – Most important features using Unsupervised to Supervised method for the clusters 0, 1, 2, and 3.

The significance of Clustering Interpretability becomes evident in scenarios where ground truth labels are absent during the development phase. This absence not only

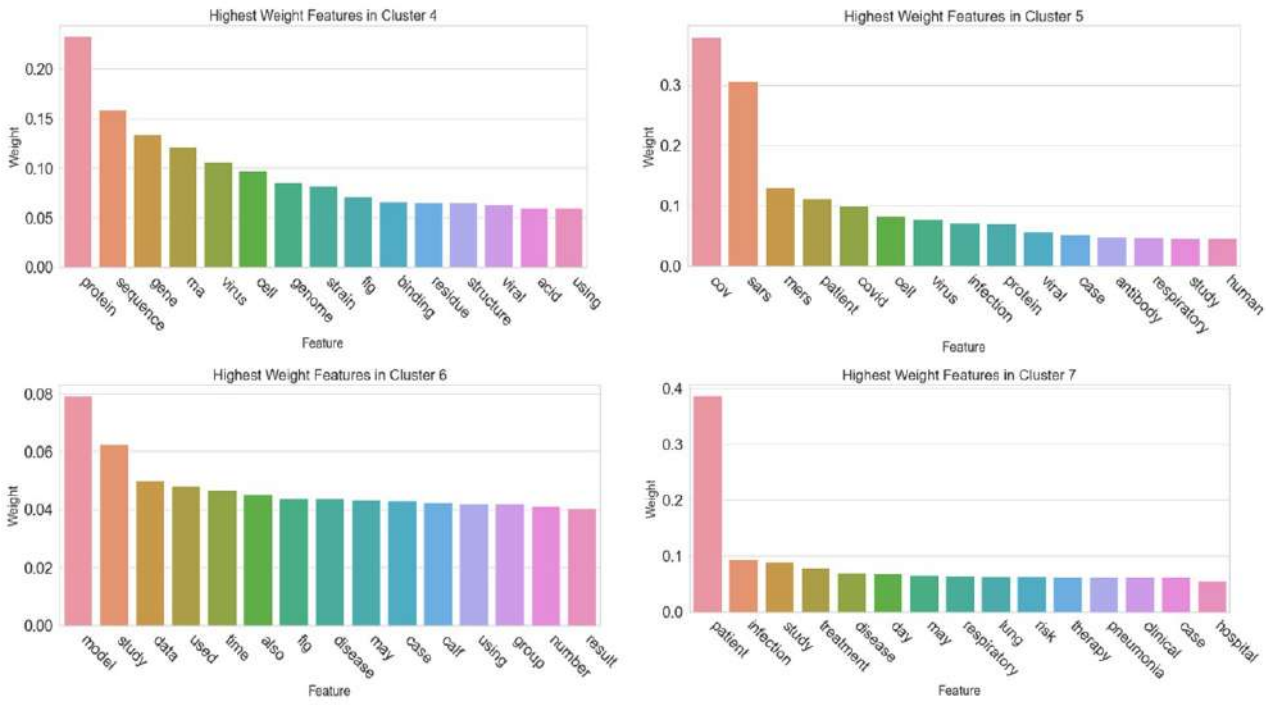

Figure 6.8 – Most important features using Unsupervised to Supervised method for the clusters 4, 5, 6, and 7.

hinders data scientists from directly assessing the validity of clustering using internal validation indices but also complicates the task of conveying cluster performance to stakeholders in a straightforward and understandable manner. In this context, we have introduced a potential method aimed at addressing this challenge. This method revolves around extracting feature importance specific to clusters, enabling us to comprehend the rationale behind the configuration of each cluster by model. This approach extends to effective communication with stakeholders and intuitive evaluation and finds applications in cluster-based Keyword Extraction within Natural Language Processing (NLP) and as a general technique for feature selection.

6.3.3.4 Topic modeling evaluation

In this experiment, we pass the outputs of our deep architecture (i.e., clusters assignment) to LDA, which extracts the topics in each document. Table 6.5 presents the topics predicted by the proposed method for each cluster. Those topics are predicted through the terms that describe each topic. In addition, Figure 6.9 shows the number of documents that have or share the same topic.

The selected topics were labeled with the names of sub-fields related to Medicine,

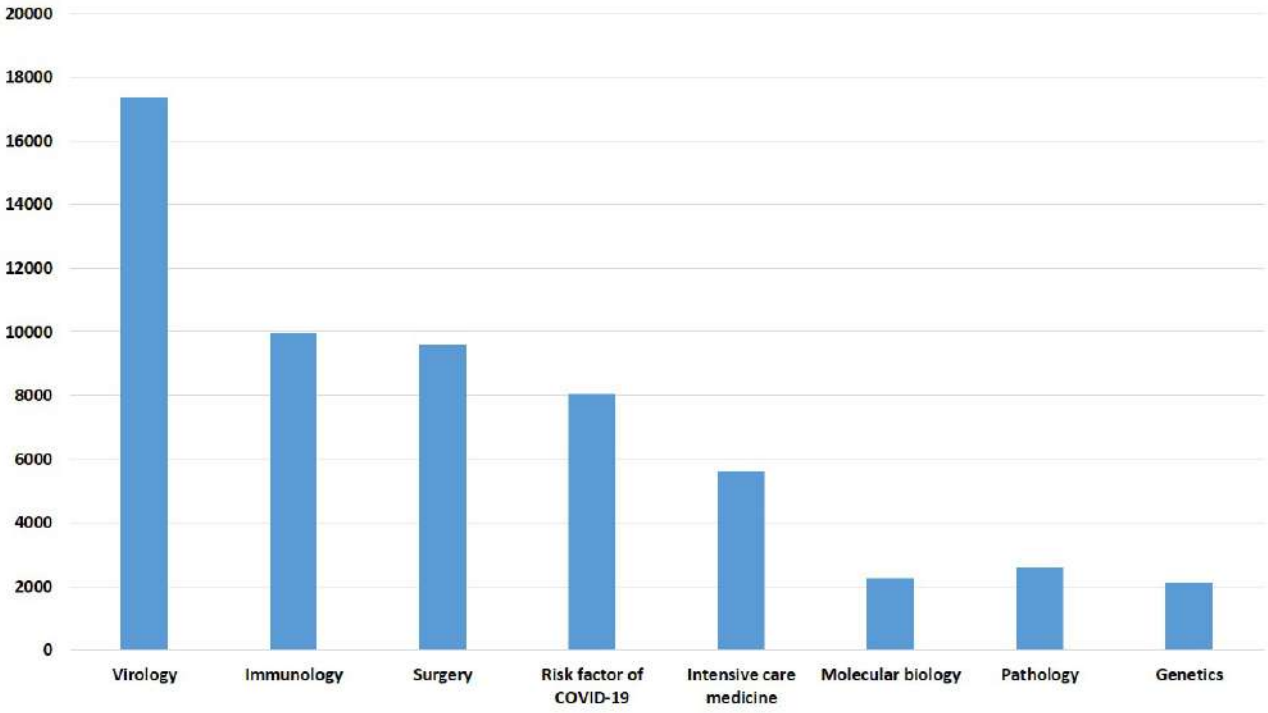

Figure 6.9 – The number of documents shared the same topics.

Biology, and Chemistry. Figure 6.9 represents the number of labeled topics: Virology, Immunology, Surgery, the Risk factor of COVID-19, Intensive Care Medicine, Molecular Biology, Pathology, and Genetics. It can be seen that the most significant number of publications is in Virology. The relationship of the sub-field of virology to COVID-19 explains this large number of publications, and this is because virology deals with the study of the COVID-19 virus, its variants, and attempts to find the appropriate vaccines. The fields of Immunology, Surgery, and Risk factors of COVID-19 have roughly equal proportions of publications. These fields are considered to be hot topics that have attracted significant attention. These fields studied the behavior and consequences of this virus on health and its impact on citizens, education, the economy, and many other areas. Finally, The least active fields during the COVID-19 pandemic are Molecular Biology, Pathology, and Genetics. The low number of publications in these areas is because these areas are not closely related to the COVID-19 pandemic.

We assess the performance of the LDA when feeding it with the entire dataset (instead of clustered documents) using the Coherence Score C_V. The aim was to measure if LDA provides a meaningful, accurate, and latent topic representation. A set of statements or facts is considered coherent if they support each other. Topic Coherence measures a single

topic's score by measuring the degree of semantic similarity between high-scoring words in the topic. These measurements help distinguish between semantically interpretable topics and topics that are artifacts of statistical inference. Figure 6.10 outlines the performance of LDA measured by C_v score while varying the number of topics.

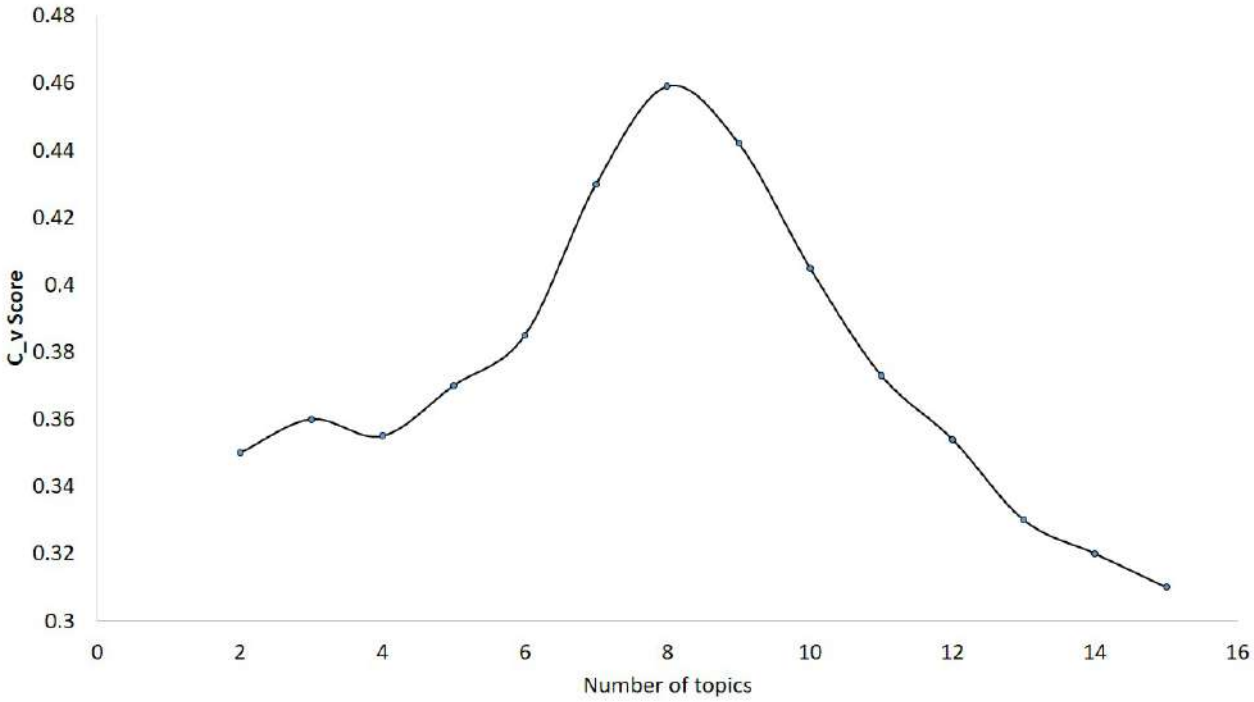

Figure 6.10 – Measuring the performance of LDA using C_v score while varying the number of topics.

The coherence score seems to keep increasing with the number of topics. In particular, the best performance is reached when the number of topics equals 8, which is analogous to the number of clusters detected by our proposed method. The difference, however, lies in the computational cost required by the two strategies. In the proposed method, the computational cost will be much less than the entire dataset as we fed the LDA with clustered documents.

6.3.3.5 Visualization

To better understand the clusters' structure and how our model organized the documents, the CORD-19 dataset was projected into a 2D space using UMAP.

Through Fig. 6.11, It is noticeable that the virology group is widespread and overlaps with all clusters, which can be explained by the relationship of the virology field with all other fields. In addition, some points appear to be outliers. However, they are clustered

in the virology field, which can be explained by how similar these points are to the points of this cluster. It can be seen that the clusters of immunology, genetics, molecular biology, and pathology overlap, and this is due to the similarity of these fields. The groups for surgery and the risk factor of COVID-19 are separated because these two fields have unique and different keywords from the rest clusters. Ultimately, it is worth mentioning that UMAP is also a dimension reduction method, and its way of work can affect the latent space produced by our model.

6.4 Conclusion

This study proposed a novel deep-learning architecture for organizing a large dataset of COVID-19-related scientific literature. The architecture comprises three main components: two encoders jointly trained with one decoder. The main idea behind our model is to train the autoencoder using a two-fold objective function that incorporates two different terms. The first term is the reconstruction loss, designed to check the latent representation, whereas the second term (clustering loss) is dedicated to clustering the input documents. Then, the Latent Dirichlet Allocation (LDA) is used to analyze the document's topics. To improve the computational efficiency of our method, we have considered feeding the LDA with the clustered documents instead of feeding the whole dataset. We conduct thorough experiments on a public dataset. Experimental results show that the proposed method can accurately predict topics. In addition, experimental results demonstrate that our method has significantly outperformed several recent studies. The next chapter concludes this dissertation and provides some perspectives.

Table 6.5 – The descriptive terms of each topic per cluster.

Topic per Cluster	Descriptive terms
Virology	coronavirus, rabies, response, gene, roost, viral, genome, sars-cov, annotation, bent-winged, herpesvirus, table, epitope, china, supplementary, microbiome, protein, sample, infection, cell, sequence, disease
Immunology	mouse, viral, antibody, immune, gene, response, culture, day, siRNA, lung, min, target, medium, membrane, patient, autophagy, apoptosis, tumor, expression, cancer, infection, bind, demyelination, tissue, receptor, effect, human, analysis, epithelial, delivery, virus, rat, type, feline, bovine, coronavirus, gut
Surgery	symptom, health, surgery, china, epidemic, virus, care, treatment, medicine, facemask, endoscopy, procedure, protein, region, humidity, -ent, tube, placement, pack, self-isolate, tip, subcutaneous, case, study, disease, risk
Risk factor of COVID-19	care, patient, hospital, disaster, outbreak, country, global, human, emergency, information, international, sars, animal, china, population, report, plan, response, research, study, train, ebola, service, infectious, pandemic, medium, nurse, individual, program, case, agent, review, management, biological, education, travel, medicine, food, air, migrant, shortlist, laboratory, surveillance, datum, risk
Intensive care medicine	medicine, pandemic, patient, hospital, symptom, health, china, epidemic, virus, care, need, woman, pregnant, infection, treatment, medicine, Chinese, facemask, endoscopy, procedure, spike, protein, region, humidity, helmet, absolute, ent, tube, placement, pack, self-isolate, tip, subcutaneous, times, case, study, disease, surgery, datum, risk, use
Molecular biology	patient, cell, antibody, lung, vaccine, case, day, mouse, hospital, treatment, drug, bind, pro, structure, activity, resistance, genotype, head, polymorphism, allele, animal, sars-cov, human, protein
Pathology	lung, virus, test, influenza, cause, antibiotic, fever, pneumonia, recipient, pulmonary, level, transplant, pathogen, exacerbation, blood, mortality, health, care, asthma, symptom, gene, bacterial, tissue, infant, child, airway, cell, lesion, signify, sequence, treatment, cancer, transport, center, wait, group, day, hospital, room
Genetics	protein, illness, hcov-, nsp, hcov-oc, cns, mouse, disorder, codon, lipid, transmission, ccov, dog, subgroup, cell, hcov, sequence, hcov-nl, genotype, child, patient, gene, strain

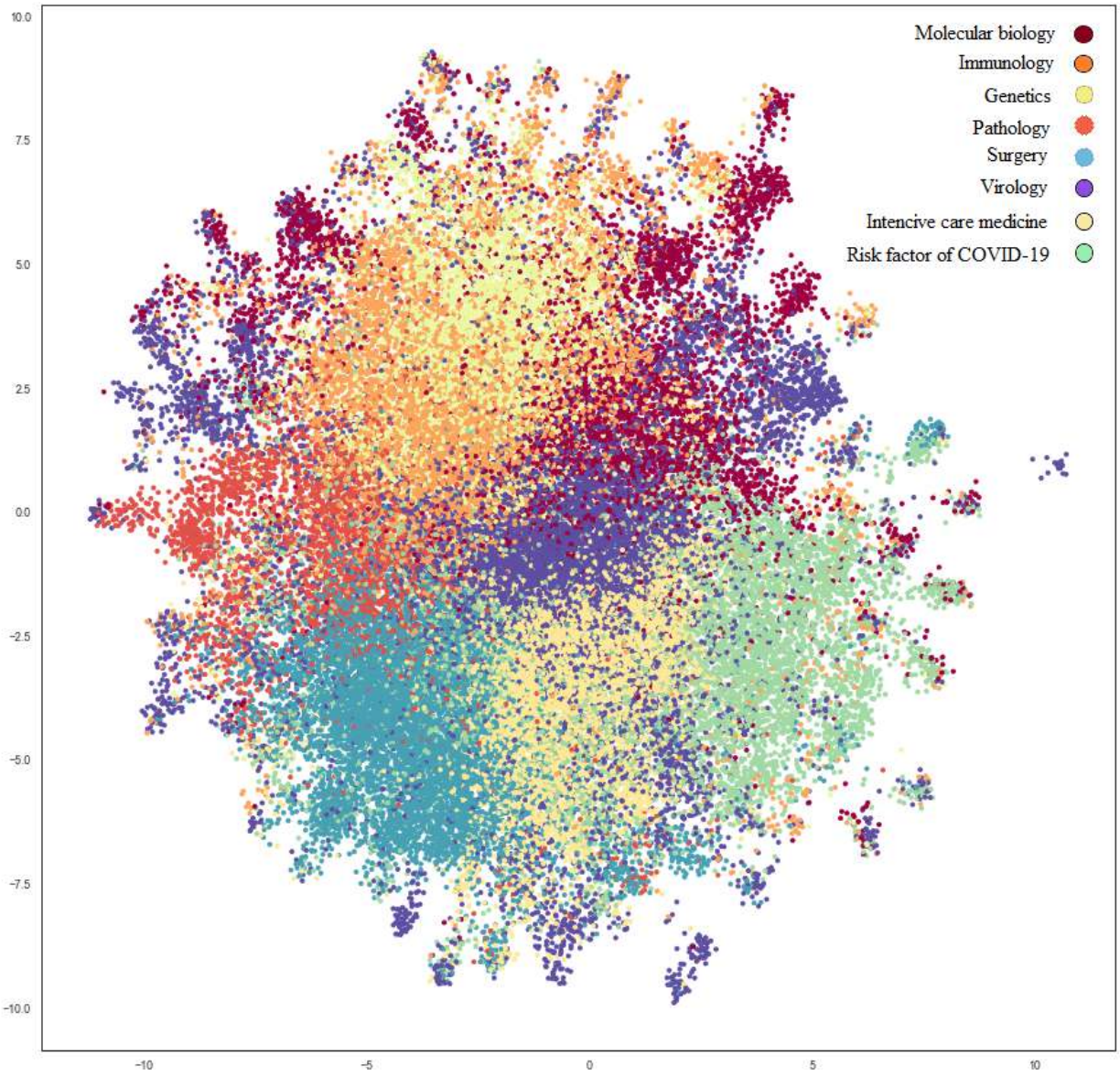

Figure 6.11 – Visualization of CORD-19 dataset using UMAP.

Chapter 7

General Conclusion and Perspectives

7.1 General Conclusion

The research addressed in this thesis centers around the significance of employing dimensionality reduction and embedding techniques in addressing various machine learning problems, with potential applications for tackling real-world challenges.

Traditional clustering algorithms often encounter significant computational challenges when confronted with high-dimensional datasets. As a result, we have incorporated UMAP, a manifold embedding technique, as a preprocessing step to enhance the performance of these clustering algorithms. The experiments conducted on the selected clustering algorithms illustrate that the proposed approach can indeed enhance their performance. As a result, these clustering algorithms are now better equipped to handle extensive datasets more efficiently and swiftly.

In this study, we introduced a novel technique called UEC (Unified Embedding and Clustering) that concurrently optimizes both data clustering and representation. UEC, a manifold-based clustering method, seeks to learn the underlying manifold within the embedding space while preserving local and global data structures. This optimization approach has resulted in the development of three distinct variants: the Comma variant, the Plus variant, and the Light Plus variant. Through empirical experiments conducted on

large-scale benchmark datasets, our performance and sensitivity analyses have illustrated the remarkable effectiveness of UEC when compared to numerous baseline algorithms.

The sheer volume of recently published scientific papers related to COVID-19 has spurred us to develop an innovative method for automatically organizing and conducting bibliometric evaluations of this literature. Our proposed approach begins by representing these documents in a numerical format, followed by clustering them, and finally, identifying the subject matter of each cluster. To achieve this, we harness some of the most advanced and sophisticated machine learning techniques, including deep autoencoders and UMAP embedding. The technology we have devised streamlines the process for researchers to sift through the vast body of relevant literature and identify notably similar papers. Moreover, cluster mapping aids in data analysis. The study's results have demonstrated the effectiveness of our approach, producing excellent outcomes.

In the same context of organizing and conducting bibliometric analysis of COVID-19-related documents, we introduce a novel deep autoencoder (DAE) architecture. This DAE architecture consists of three primary components: two encoders trained jointly with a single decoder. The autoencoder is primarily trained through a two-fold objective function that encompasses two distinct components. The first component is engineered to ensure the quality of the latent representation, while the second component is focused on clustering the input documents. Subsequently, the topics within the documents are analyzed using Latent Dirichlet Allocation (LDA). Our proposed method demonstrates precise topic predictions based on experimental data. Moreover, the empirical results indicate that our approach surpasses recent research significantly in terms of performance.

7.2 Perspectives

While the proposed approaches have showcased impressive performance in hand modalities recognition, there remains room for further development and improvement in future research endeavors.

1. In our ongoing research, we plan to assess the performance of our first proposed method by applying it to other extensive datasets. Additionally, we intend to explore the utilization of various clustering and dimensionality reduction techniques,

particularly those associated with deep and manifold learning, to enhance our approach further.

2. Regarding the UEC technique, our future efforts will be directed toward refining the multi-objective function to enhance the algorithm's performance in terms of results and computational efficiency. Furthermore, we plan to delve into exploring the impact of various embedding methods on the performance of the clustering task.
3. We can evaluate and refine our approach by testing the effect of loss functions and distance measures on our technique. By comparing different loss functions and distance measures, we can gain insights into how they impact the performance and behavior of our technique.
4. For the COVID-19-related contributions, We plan to use the techniques presented as the foundation for a search engine. The user could obtain highly particular documents from the dataset by submitting queries or inquiries.
5. We can investigate the performance of the other well-suited techniques for Natural language processing and topic modeling, e.g., Long Short Term Memory (LSTM).

Personal Contributions

Journal Publications

— Allaoui, M., Kherfi, M. L., Cheriet, A., and Bouchachia, A. (2023). Unified Embedding and Clustering. Expert Systems with Applications. DOI: https://doi.org/10.1016/j.eswa.2023.121923

— Allaoui, M., Kherfi, M. L., Aiadi, O., and Belhaouari, S. B. (2023). A novel two-fold loss function for data clustering and reconstruction: application to document analysis. IEEE Access. DOI: 10.1109/ACCESS.2023.3312622

International Communications

— Allaoui, M., Kherfi, M. L., and Cheriet, A. (2020, June). Considerably improving clustering algorithms using UMAP dimensionality reduction technique: A comparative study. In International conference on image and signal processing (pp. 317-325). Cham: Springer International Publishing. DOI https://doi.org/10.1007/978-3-030-51935-3_34

— Allaoui, M., Aissa, N. E. H. S. B., Belghith, A. B., and Kherfi, M. L. (2021, September). A machine learning-based tool for exploring covid-19 scientific literature. In 2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI) (pp. 1-7). IEEE. DOI: 10.1109/ICRAMI52622.2021.9585958

Part III

Appendices

Appendix A

Appendix A: The derivation of the embedding objective function (Cross-Entropy)

The Cross-Entropy function depends only on z_i, so in this part, we derive the CE w.r.t. z_i. For The partial derivative of the CE w.r.t. z_i, we used the same derivative of UMAP.

Let us first compute the derivative of q_{ij} and $1-q_{ij}$ we have:

$$q_{ij} = (1 + a\left\|z_i - z_j\right\|_2^{2b})^{-1} \tag{A.1}$$

$$\frac{\partial q_{ij}}{\partial z_i} = \frac{-2ab\left\|z_i - z_j\right\|_2^{2b-1}}{(1 + a\left\|z_i - z_j\right\|_2^{2b})^2} \tag{A.2}$$

Now we can derive the CE objective function as follows:

$$CE(P,Q) = \sum_i \sum_j \left[p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}}) \right]$$

$$= \sum_i \sum_j \Big[p_{ij} \log(p_{ij}) - p_{ij} \log(q_{ij}) + (1-p_{ij}) \log(1-p_{ij}) - (1-p_{ij}) \log(1-q_{ij}) \Big]$$

Since p_{ij} doesn't depend on z_i, the derivatives of $p_{ij}\log(p_{ij})$ and $(1-p_{ij})\log(1-p_{ij})$ are zero, so we can derive the CE as follow:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[-\frac{\partial q_{ij}}{\partial z_i} \frac{p_{ij}}{q_{ij}} - \frac{\partial (1-q_{ij})}{\partial z_i} \frac{1-p_{ij}}{1-q_{ij}} \right] \tag{A.3}$$

We have $1-q_{ij} = \frac{a\left\|z_i - z_j\right\|_2^{2b}}{1+a\left\|z_i - z_j\right\|_2^{2b}}$, now we will substitute the last one, Eq. A.1 and Eq. A.2 in the Eq. A.3 so we can rewrite it as follow:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[-\frac{-2ab\left\|z_i - z_j\right\|_2^{2b-1}}{(1+a\left\|z_i - z_j\right\|_2^{2b})^2} \frac{p_{ij}}{(1+a\left\|z_i - z_j\right\|_2^{2b})^{-1}} - \frac{2ab\left\|z_i - z_j\right\|_2^{2b-1}}{(1+a\left\|z_i - z_j\right\|_2^{2b})^2} \frac{1-p_{ij}}{\frac{a\left\|z_i - z_j\right\|_2^{2b}}{1+a\left\|z_i - z_j\right\|_2^{2b}}} \right]$$

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2ab\left\|z_i - z_j\right\|_2^{2(b-1)} p_{ij}}{1+a\left\|z_i - z_j\right\|_2^{2b}} - \frac{2b(1-p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1+a\left\|z_i - z_j\right\|_2^{2b})} \right] \left\|z_i - z_j\right\|$$

Since the exponential power is taking a long calculation time, so we can reduce this term $\frac{2ab\left\|z_i - z_j\right\|_2^{2(b-1)}}{1+a\left\|z_i - z_j\right\|_2^{2(b)}}$ as shown below:

$$\frac{2ab\left\|z_i - z_j\right\|_2^{2(b-1)}}{1+a\left\|z_i - z_j\right\|_2^{2(b)}} = \frac{2b}{1/a\left\|z_i - z_j\right\|_2^{2(b-1)} + \left\|z_i - z_j\right\|_2^2}$$

Now we can write the derivative of the embedding loss function as follows:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/a\left\|z_i - z_j\right\|_2^{2(b-1)} + \left\|z_i - z_j\right\|_2^2} - \frac{2b(1-p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1 + a\left\|z_i - z_j\right\|_2^{2b})} \right] \left\|z_i - z_j\right\| \quad \text{(A.4)}$$

Appendix B

Plus variant derivations

We describe how the clustering loss function is derived considering the auxiliary target distribution T_{ik}, which depends on z_i and μ_k. First let us recall that S_{ik} and T_{ik} are expressed as follows:

$$S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1} \tag{B.1}$$

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})} \tag{B.2}$$

The derivative of the objective function F w.r.t. z_i:

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \tag{B.3}$$

The derivation of the cross entropy is given in A, while the derivation of the KL divergence w.r.t. z_i is given in the following.

$$KL(T||S) = \sum_i \sum_k [T_{ik} \log T_{ik} - T_{ik} \log S_{ik}]$$

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i} \log T_{ik} + T_{ik} \frac{\frac{\partial T_{ik}}{\partial z_i}}{T_{ik}} - \frac{\partial T_{ik}}{\partial z_i} \log S_{ik} - \frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}} \right]$$

$$+ \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i} \log T_{i'k} + T_{i'k} \frac{\frac{\partial T_{i'k}}{\partial z_i}}{T_{i'k}} - \frac{\partial T_{i'k}}{\partial z_i} \log S_{i'k} - \frac{\partial S_{i'k}}{\partial z_i} \frac{T_{i'k}}{S_{i'k}} \right]$$

The derivative is formulated as a sum of two parts since T_{ik} needs to compute its

derivative w.r.t. z_i according to two cases. The first is when $i' = i$ and the second is when $i' \neq i$. Since i' is different from i, $S_{i'k}$ does not depend on z_i (see Eq. B.1), $\frac{\partial S_{i'k}}{\partial z_i} = 0$. We can simplify the derivative as follows:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}} \right] + \sum_{i' \neq i} \sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1 + \log \frac{T_{i'k}}{S_{i'k}}) \right] \quad \text{(B.4)}$$

The derivative of S_{ik} w.r.t. z_i is:

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2}$$

We can write $\frac{\partial S_{ik}}{\partial z_i}$ as follows:

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2b}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} S_{ik} \quad \text{(B.5)}$$

We must distinguish two cases to derive T_{ik}. The first one corresponds to $i' = i$ (written as T_{ik}), and the second case corresponds to $i' \neq i$ (written as $T_{i'k}$). The expression of T_{ik} and $T_{i'k}$ is the same as in Eq. B.2.

$$T_{ik} = \frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})}$$

$$T_{i'k} = \frac{S_{i'k}/\sum_l S_{lk}}{\sum_m (S_{i'm}/\sum_l S_{lm})}$$

Let us show the derivation of T_{ik} w.r.t. z_i by considering the numerator and the denominator separately. The numerator is given as:

$$N_{ik} = \frac{S_{ik}}{\sum_l S_{lk}} \quad \text{(B.6)}$$

and its derivative is:

$$\frac{\partial N_{ik}}{\partial z_i} = \frac{\frac{\partial S_{ik}}{\partial z_i}\sum_l S_{lk} - S_{ik}\frac{\partial \sum_l S_{lk}}{\partial z_i}}{(\sum_l S_{lk})^2}$$

Since S_{lk} depends on z_i in the only case where $l = i$, $\frac{\partial \sum_l S_{lk}}{\partial z_i} = \sum_l \frac{\partial S_{lk}}{\partial z_i}$, where $\frac{\partial S_{lk}}{\partial z_i} =$

$\frac{\partial S_{ik}}{\partial z_i}$ only when $l = i$ and $\frac{\partial S_{lk}}{\partial z_i} = 0$, where $l \neq i$. Now we can write $\frac{\partial N_{ik}}{\partial z_i}$ as follows:

$$\frac{\partial N_{ik}}{\partial z_i} = \frac{\frac{\partial S_{ik}}{\partial z_i} \sum_{l \neq i} S_{lk}}{(\sum_l S_{lk})^2} \tag{B.7}$$

For the denominator, which is expressed as:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}} \tag{B.8}$$

the derivative w.r.t. z_i is given as follows:

$$\frac{\partial D_i}{\partial z_i} = \sum_m \frac{\frac{\partial S_{im}}{\partial z_i} \sum_l S_{lm} - S_{im} \frac{\partial \sum_l S_{lm}}{\partial z_i}}{(\sum_l S_{lm})^2}$$

In the same way and for all m, $\frac{\partial \sum_l S_{lm}}{\partial z_i} = \frac{\partial S_{im}}{\partial z_i} = -\frac{2ab(z_i - \mu_m)^{2b-1}}{(1+a(z_i-\mu_m)^{2b})^2}$. Consequently:

$$\frac{\partial D_i}{\partial z_i} = \sum_m \frac{\frac{\partial S_{im}}{\partial z_i} \sum_{l \neq i} S_{lm}}{(\sum_l S_{lm})^2} \tag{B.9}$$

The derivative of T_{ik} w.r.t. z_i is:

$$\frac{\partial T_{ik}}{\partial z_i} = \frac{\frac{\partial N_{ik}}{\partial z_i} D_i - N_{ik} \frac{\partial D_i}{\partial z_i}}{D_i^2} \tag{B.10}$$

where $N_{ik}, \frac{\partial N_{ik}}{z_i}, D_i, \frac{\partial D_i}{z_i}$ are substituted by Eqs. B.6, B.7, B.8, and B.9 respectively in Eq.B.10.

We compute the derivative of $T_{i'k}$ w.r.t. z_i. In Eq. B.4, we use $T_{i'k}$ to denote the case where $i' \neq i$.. The numerator and the denominator are derived separately. Like in Eq. B.6, the numerator is given:

$$N_{i'k} = \frac{S_{i'k}}{\sum_l S_{lk}} \tag{B.11}$$

Since $S_{i'k}$ does not depend on z_i, its derivative is 0. Also $\frac{\partial \sum_l S_{lk}}{\partial z_i} = \sum_l \frac{\partial S_{lk}}{\partial z_i}$ such that $\frac{\partial S_{lk}}{\partial z_i} = \frac{\partial S_{ik}}{\partial z_i}$ only when $l = i$ and $\frac{\partial S_{lk}}{\partial z_i} = 0$ otherwise. The derivative of N_{ik} w.r.t. z_i is:

$$\frac{\partial N_{i'k}}{\partial z_i} = \frac{-S_{i'k} \frac{\partial S_{ik}}{\partial z_i}}{(\sum_l S_{lk})^2} \tag{B.12}$$

where $\frac{\partial S_{ik}}{\partial z_i}$ is given by Eq. B.5.

Like in Eq. B.7, the denominator is given as follows:

$$D_{i'} = \sum_m \frac{S_{i'm}}{\sum_l S_{lm}} \tag{B.13}$$

Since $S_{i'm}$ does not depend on z_i, its derivative is 0. $\frac{\partial \sum_l S_{lm}}{\partial z_i} = \sum_l \frac{\partial S_{lm}}{\partial z_i}$ where $\frac{\partial S_{lm}}{\partial z_i} = \frac{\partial S_{im}}{\partial z_i}$ only when $l = i$ such that $\frac{\partial S_{im}}{\partial z_i} = -\frac{2ab\|z_i-\mu_k\|_2^{2b-1}}{(1+a\|z_i-\mu_k\|_2^{2b})^2}$ and $\frac{\partial S_{lm}}{\partial z_i} = 0$ otherwise. We can write the derivative of D as follows:

$$\frac{\partial D_{i'}}{\partial z_i} = \sum_m \frac{-S_{i'm}\frac{\partial S_{im}}{\partial z_i}}{(\sum_l S_{lm})^2} \tag{B.14}$$

Now the derivative of $T_{i'k}$ w.r.t. z_i is:

$$\frac{\partial T_{i'k}}{\partial z_i} = \frac{\frac{\partial N_{i'k}}{\partial z_i} D_{i'} - N_{i'k}\frac{\partial D_{i'}}{\partial z_i}}{D_{i'}^2} \tag{B.15}$$

where $N_{i'k}$, $\frac{\partial N_{i'k}}{z_i}, D_{i'}, \frac{\partial D_{i'}}{z_i}$ are substituted by Eqs. B.11, B.12, B.13, B.14 respectively in Eq.B.15.

Now let's recall the derivative of the KL-divergence, Eq. B.4:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}}\right] + \sum_{i'\neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right]$$

We substitute S_{ik} and $\frac{\partial S_{ik}}{\partial z_i}$ by Eqs.B.1 and B.5 in Eq. B.4. to obtain:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) + \frac{2bS_{ik}}{1/\|z_i-\mu_k\|_2^{2b-1} + \|z_i-\mu_k\|_2^2}\frac{T_{ik}}{(1+a\|z_i-\mu_k\|_2^{2b})^{-1}}\right] + \sum_{i'\neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right]$$

In Eq. B.4, The term $\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}}$ can be more simpler as follow:

$$\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}} = -\frac{2bT_{ik}}{1/\|z_i-\mu_k\|_2^{2b-1}+\|z_i-\mu_k\|_2^2}$$

Substituting $\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}}$ in $\frac{\partial KL}{\partial z_i}$ by its expression, we obtain the final formulation of the derivative of KL:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i-\mu_k\|_2^{2b-1}+\|z_i-\mu_k\|_2^2}\right] + \sum_{i'\neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right] \quad \text{(B.16)}$$

B.1 Derivative of the total loss function w.r.t. datapoints

Now, we assemble the two parts by substituting the derivative of Cross Entropy (Eq. A.4) and the derivative of KL divergence (Eq. B.16) in the derivative of the total loss function to obtain:

$$\frac{\partial F}{\partial \mu_k} = \alpha\frac{\partial CE}{\partial \mu_k} + \beta\frac{\partial KL}{\partial \mu_k}$$

$$\frac{\partial F}{\partial z_i} = \alpha\sum_j \left[\frac{2bp_{ij}}{1/a\left\|z_i-z_j\right\|_2^{2(b-1)}+\left\|z_i-z_j\right\|_2^2} - \frac{2b(1-p_{ij})}{\left\|z_i-z_j\right\|_2^2(1+a\left\|z_i-z_j\right\|_2^{2b})}\right]\left\|z_i-z_j\right\|_2 + \beta\left(\sum_k\left[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i-\mu_k\|_2^{2b-1}+\|z_i-\mu_k\|_2^2}\right] + \sum_{i'\neq i}\sum_k\left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right]\right) \quad \text{(B.17)}$$

The update of z_i is performed using $\frac{\partial F}{\partial z_i}$ where α and β are parameters to be set.

However, Eq. B.17 refers to the two fractions: $\log \frac{T_{ik}}{S_{ik}}$ and $\log \frac{T_{i'k}}{S_{i'k}}$ which could involve division by 0.

To solve this problem, we do some studies and analysis on the definition domain of S_{ik} and T_{ik}, the sign of hyper-parameter a, and the definition domain of $\log \frac{T_{ik}}{S_{ik}}$.

The sign of hyper-parameter 'a'

Define $\phi : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$, a smooth approximation of the membership strength between two points in $\mathbb{R}^d$:

$$\phi(z_i, z_j) = q_{ij} = (1 + \left\| z_i - z_j \right\|_2^{2b})^{-1}$$

where 'a' and 'b' are chosen by non-linear least square fitting against the curve $\psi : \mathbb{R} \times \mathbb{R} \to [0,1]$ where:

$$\psi(z_i, z_j) \quad = \quad \begin{cases} 1 & if \left\| z_i - z_j \right\|_2 \leq \text{min-dist} \\ \exp(-(\left\| z_i - z_j \right\|_2 - \text{min-dist})) & \text{otherwise} \end{cases}$$

So the sign of 'a' is always positive since $q_{ij} \in [0,1]$ and that is mean:

$$q_{ij} \to \begin{cases} 0 & if \;\; 1 + a\left\| z_i - z_j \right\|_2^{2b} \to +\infty \\ 1 & if \;\; a\left\| z_i - z_j \right\|_2^{2b} = 0 \end{cases}$$

So since $\left\| z_i - z_j \right\|_2$ is always positive, we conclude that 'a' is also always positive.

Definition domain of S_{ik} and T_{ik}

Define $S_{ik} : \mathbb{R}^d \times \mathbb{R}^d \to [0,1]$, a soft assignment between the datapoints and the cluster centroid in $\mathbb{R}^d$:

$$S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1} = \frac{1}{1 + a\|z_i - \mu_k\|_2^{2b}}$$

$$S_{ik} \text{ is defined since} \Leftrightarrow \begin{cases} \|z_i - \mu_k\|_2^{2b} & \text{is defined} \\ a\|z_i - \mu_k\|_2^{2b} \neq -1 \end{cases}$$

$$\Leftrightarrow \begin{cases} \|z_i - \mu_k\|_2^{2b} & \text{is always defined} \\ a \neq \frac{-1}{\|z_i - \mu_k\|_2^{2b}} & \text{is defined since } a \text{ is positive} \end{cases}$$

Now since T_{ik} is based on S_{ik} so it is also defined on the interval $[0, 1]$.

Definition domain of $\log \frac{T_{ik}}{S_{ik}}$

Let us set $g(z_i, \mu_k) = \log \frac{T_{ik}}{S_{ik}}$

$$\text{g is defined if and only if} \Leftrightarrow \begin{cases} S_{ik} \neq 0 \quad (1) \\ \frac{T_{ik}}{S_{ik}} > 0 \end{cases} \tag{B.18}$$

Where:

$$\frac{T_{ik}}{S_{ik}} > 0 \Leftrightarrow \begin{cases} T_{ik} \neq 0 & (2) \\ S_{ik} \text{ and } T_{ik} \text{ has the same sign} & (3) \end{cases} \tag{B.19}$$

As we see in the previews Eq. B.18 we have three conditions let us treat them separately. The first condition is satisfied if and only if:

$$(1) \Leftrightarrow \frac{1}{1+a\left\|z_i - z_j\right\|_2^{2b}} \nrightarrow 0 \Leftrightarrow \begin{cases} 1+a\left\|z_i - z_j\right\|_2^{2b} \nrightarrow \infty \\ 1+a\left\|z_i - z_j\right\|_2^{2b} \neq 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} a\left\|z_i - z_j\right\|_2^{2b} \nrightarrow \infty \quad (*) \\ a\left\|z_i - z_j\right\|_2^{2b} \neq -1 \quad \text{is always satisfied} \end{cases}$$

$$(*) \Leftrightarrow a\left\|z_i - z_j\right\|_2^{2b} \nrightarrow +\infty \text{ since } -\infty \text{ is excluded}$$
$$\Leftrightarrow \left\|z_i - z_j\right\|_2^{2b} \nrightarrow +\infty$$

The cases where $\left\|z_i - z_j\right\|_2^{2b} \to +\infty$ are:

$$\left\|z_i - z_j\right\|_2^{2b} \to +\infty \Leftrightarrow \begin{cases} \left\|z_i - z_j\right\|_2^{2b} \to +\infty \text{ and } b \geq 0 \text{ (A)} \\ \left\|z_i - z_j\right\|_2^{2b} \to 0 \text{ and } b < 0 \text{ (B)} \end{cases}$$

Now to make **condition (1)** $S_{ik} \nrightarrow 0$ satisfied it should make conditions (A) and (B) not satisfied.

Now we pass to **condition (2)**: $T_{ik} \neq 0$

$$T_{ik} = \frac{S_{ik} / \sum_l S_{lk}}{\sum_m (S_{im} / \sum_l S_{lm})}$$

So T_{ik} is defined and $T_{ik} = 0$ if $S_{ik} = 0$.

Condition (3): T_{ik} and S_{ik} have the same sign and this condition is always satisfied since $S_{ik} \in [0, 1]$ and T_{ik} is computed based on S_{ik} so they have always the same sign.

From what we mentioned above, we can conclude our solutions to solve the division problem by zero as follows. The first one is shown in algorithm 2:

Algorithm 2 Solving the problem of division by zero

if $b \geq 0$ **then**
  if $\|z_i - \mu_k\|_2 > Dist - Max$ **then**
    we have two solutions:
    Consider z_i as an outlier
    Replace $\|z_i - \mu_k\|_2$ by Dist-Max
  end if
else
  Solve the cases where $z_i \approx \mu_k$
end if

And the second solution is as follows:

$$\frac{T_{ik}}{S_{ik}} = \frac{\frac{S_{ik}/\sum_l S_{lk}}{\sum_m (S_{im}/\sum_l S_{lm})}}{S_{ik}}$$
$$= \frac{1}{(\sum_l S_{lk})(\sum_m (S_{im}/\sum_l S_{lm}))}$$

B.2 The derivative of Objective Function w.r.t. the centroids

Now, we will drive our objective function w.r.t. μ_k. First, let's recall our weighted sum function:

$$F(P,Q,T,S) = \alpha CE(P,Q) + \beta KL(T||S) \tag{B.20}$$

Such that:

$$CE(P,Q) = \sum_i \sum_j \left[p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}}) \right]$$

$$KL(T||S) = \sum_i \sum_k T_{ik} \log \frac{T_{ik}}{S_{ik}}$$
$$= \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik}$$

Also, let's recall the derivative of our weighted sum function:

$$\frac{\partial F}{\partial \mu_k} = \alpha \frac{\partial CE}{\partial \mu_k} + \beta \frac{\partial KL}{\partial \mu_k}$$

CE doesn't depend on cluster centroids, so its derivative is zero. Therefore, we only compute the derivative of the KL divergence w.r.t. μ_k:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} \log T_{ik} + T_{ik} \frac{\frac{\partial T_{ik}}{\partial \mu_k}}{T_{ik}} - \frac{\partial T_{ik}}{\partial \mu_k} \log S_{ik} - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} \log T_{ik'} + T_{ik'} \frac{\frac{\partial T_{ik'}}{\partial \mu_k}}{T_{ik'}} - \frac{\partial T_{ik'}}{\partial \mu_k} \log S_{ik'} - \frac{\partial S_{ik'}}{\partial \mu_k} \frac{T_{ik'}}{S_{ik'}} \right]$$

We divided the derivative into a sum of 2 parts since T_{ik} needs to compute its derivative with respect μ_k in two cases. The first is when $k' = k$, and the second is when $k' \neq k$. Also, since $S_{ik'}$ doesn't depend on μ_k so $\frac{\partial S_{ik'}}{\partial \mu_k} = 0$.

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} \log T_{ik} + \frac{\partial T_{ik}}{\partial \mu_k} - \frac{\partial T_{ik}}{\partial \mu_k} \log S_{ik} - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} \log T_{ik'} + \frac{\partial T_{ik'}}{\partial \mu_k} - \frac{\partial T_{ik'}}{\partial \mu_k} \log S_{ik'} \right]$$

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} (1 + \log \frac{T_{ik'}}{S_{ik'}}) \right] \quad \text{(B.21)}$$

To simplify the derivative of the KL divergence w.r.t. μ_k we calculate the derivatives of $S_{ik} = (1 + a\|z_i - \mu_k\|_2^{2b})^{-1}$ and T_{ik} w.r.t. μ_k in case of $k^{'} = k$. In addition, we compute the derivatives of $S_{ik'}$ and $T_{ik'}$ w.r.t. μ_k in the case $k' \neq k$. Now we start by $\frac{\partial S_{ik}}{\partial \mu_k}$

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2bS_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \quad \text{(B.22)}$$

And since $S_{ik'}$ doesn't depend on μ_k, so we can conclude that

$$\frac{\partial S_{ik'}}{\partial \mu_k} = 0 \tag{B.23}$$

Also here we should derive T_{ik} when $k' = k$ and $T_{ik'}$ when $k' \neq k$ w.r.t. μ_k. Now we are going to start with T_{ik}:

$$T_{ik} = \frac{S_{ik} / \sum_l S_{lk}}{\sum_m (S_{im} / \sum_l S_{lm})}$$

let's derive the numerator and the denominator separately. We have the numerator is:

$$N_{ik} = \frac{S_{ik}}{\sum_l S_{lk}}$$

So the derivative of N_{ik} w.r.t. μ_k is:

$$\frac{\partial N_{ik}}{\partial \mu_k} = \frac{\frac{\partial S_{ik}}{\partial \mu_k} \sum_l S_{lk} - S_{ik} \sum_l \frac{\partial S_{lk}}{\partial \mu_k}}{(\sum_l S_{lk})^2}$$

such that $\frac{\partial S_{ik}}{\partial z_i}$ is given by Eq. B.22 and $\frac{\partial S_{lk}}{\partial z_i}$ is deduced from Eq. B.22 as follows:

$$\frac{\partial S_{lk}}{\partial \mu_k} = \frac{2b S_{lk}}{1 / \|z_l - \mu_k\|_2^{2b-1} + \|z_l - \mu_k\|_2^2}$$

We have the denominator is:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}}$$

We have $\frac{\partial \sum_m \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k} = \sum_m \frac{\partial \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k}$ where $\frac{\partial \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k} \neq 0$ when $m = k$. And $\frac{\partial \frac{S_{im}}{\sum_l S_{lm}}}{\partial \mu_k} = 0$ when $m \neq k$, since $\frac{\partial S_{im}}{\partial \mu_k} = \frac{\partial S_{ik}}{\partial \mu_k} = \frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2}$ when $m = k$, and $\frac{\partial S_{im}}{\partial \mu_k} = 0$ otherwise. Also $\sum_l \frac{\partial S_{lm}}{\partial \mu_k} = \sum_l \frac{\partial S_{lk}}{\partial \mu_k} = \sum_l \frac{2ab\|z_l - \mu_k\|_2^{2b-1}}{(1 + a\|z_l - \mu_k\|_2^{2b})^2}$ when $m = k$, and $\sum_l \frac{\partial S_{lm}}{\partial \mu_k} = 0$ otherwise. So we can write $\frac{\partial D_i}{\partial \mu_k}$ as follow:

$$\frac{\partial D_i}{\partial \mu_k} = \frac{\frac{\partial S_{ik}}{\partial \mu_k} \sum_l S_{lk} - S_{ik} \sum_l \frac{\partial S_{lk}}{\partial \mu_k}}{(\sum_l S_{lk})^2}$$

We can see that $\frac{\partial N_{ik}}{\partial \mu_k} = \frac{\partial D_i}{\partial \mu_k}$. So the derivative of T_{ik} w.r.t. μ_k is:

$$\frac{\partial T_{ik}}{\partial \mu_k} = \frac{\frac{\partial N_{ik}}{\partial \mu_k}(D_i - N_{ik})}{D_i^2}$$

We do the same thing for the derivative of $T_{ik'}$ w.r.t. μ_k in the case here is $k' \neq k$. Let us first write $T_{ik'}$:

$$T_{ik'} = \frac{S_{ik'} / \sum_l S_{lk'}}{\sum_m (S_{im} / \sum_l S_{lm})}$$

we are going to derive the numerator and denominator separately:

$$N_{ik'} = \frac{S_{ik'}}{\sum_l S_{lk'}}$$

Since $S_{ik'}$ doesn't depend on μ_k, we can deduce that $\frac{\partial S_{ik'}}{\partial \mu_k} = 0$ from Eq. B.1 and B.22. Also for $\sum_l S_{lk'}$ doesn't depend on μ_k so $\sum_l \frac{\partial S_{lk'}}{\partial \mu_k} = 0$. We can conclude that the derivative of $N_{ik'}$ w.r.t. μ_k is zero. Let us move now to the denominator:

$$D_i = \sum_m \frac{S_{im}}{\sum_l S_{lm}}$$

We already computed the derivative of D_i, so let us just recall it:

$$\frac{\partial D_i}{\partial \mu_k} = \frac{\frac{\partial S_{ik}}{\partial \mu_k} \sum_l S_{lk} - S_{ik} \sum_l \frac{\partial S_{lk}}{\partial \mu_k}}{(\sum_l S_{lk})^2}$$

And from previous equations, we can conclude that the derivative of $T_{ik'}$ is as follows:

$$\frac{\partial T_{ik'}}{\partial \mu_k} = \frac{-N_{ik'} \frac{\partial D_i}{\partial \mu_k}}{D_i^2}$$

Now we are going to substitute the previews equations in the derivative of the KL divergence Eq. B.21 w.r.t. μ_k:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} (1 + \log \frac{T_{ik'}}{S_{ik'}}) \right]$$

We substitute S_{ik} and $\frac{\partial S_{ik}}{\partial \mu_k}$ by Eq. B.1 and Eq. B.22, so let's recall $\frac{\partial S_{ik}}{\partial \mu_k}$:

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2bS_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}$$

Then we are going to multiply it by $\frac{T_{ik}}{S_{ik}}$ in order to get a simpler form:

$$\frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} = \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2}$$

Substituting $\frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}}$ in $\frac{\partial KL}{\partial \mu_k}$ by the last equation, we obtain:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i \left[\frac{\partial T_{ik}}{\partial \mu_k} (1 + \log \frac{T_{ik}}{S_{ik}}) - \frac{2bT_{ik}}{1/\|z_i - \mu_k\|_2^{2b-1} + \|z_i - \mu_k\|_2^2} \right] + \sum_i \sum_{k' \neq k} \left[\frac{\partial T_{ik'}}{\partial \mu_k} (1 + \log \frac{T_{ik'}}{S_{ik'}}) \right]$$

Appendix C

Light plus variant derivations

Our optimisation purpose is to update the datapoints z_i and the centroids μ_k, so first, we derived the function F w.r.t. z_i. Then, we compute the derivative of the weighted sum function w.r.t. μ_k. However, the CE function does not depend on the cluster centres so we compute only the derivative of the clustering loss function. Now for the derivative of the weighted sum function we have:

$$\frac{\partial F}{\partial z_i} = \alpha \frac{\partial CE}{\partial z_i} + \beta \frac{\partial KL}{\partial z_i} \tag{C.1}$$

C.1 Derivation of the KL divergence w.r.t. datapoints

Notice that for the datapoints z_i we have two objective functions that need to compute their derivations w.r.t. z_i. The derivation of the CE function is already explained in A. In this part, we provide the details of the derivative of the KL divergence w.r.t. z_i. Let's recall the KL divergence:

$$KL(T, S) = \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik} \tag{C.2}$$

When updating z_i, T_{ik} is already computed and is considered as a constant number, T_{ik} doesn't depend on z_i, thus the derivative of $T_{ik} \log T_{ik}$ w.r.t. z_i is zero. We obtain:

$$\frac{\partial KL}{\partial z_i} = \sum_k -\frac{\partial S_{ik}}{\partial z_i} \frac{T_{ik}}{S_{ik}} \tag{C.3}$$

We can derived S_{ik} w.r.t. z_i as follow:

$$\frac{\partial S_{ik}}{\partial z_i} = -\frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2} \tag{C.4}$$

Substituting S_{ik} and its derivative $\frac{\partial S_{ik}}{\partial z_i}$ (Eq. C.4) in Eq. C.3. we can conclude $\frac{\partial KL}{\partial z_i}$ like this:

$$\frac{\partial KL}{\partial z_i} = \sum_k -\frac{\partial S_{ik}}{\partial z_i}\frac{T_{ik}}{S_{ik}}$$

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{C.5}$$

C.2 Derivative of the total loss function w.r.t. datapoints

Now, we assemble the two parts by substituting the derivative of Cross Entropy (Eq. A.4), and the derivative of KL divergence (Eq. C.5) in the total loss function (Eq. C.1) to obtain:

$$\frac{\partial F}{\partial z_i} = \alpha\frac{\partial CE}{\partial z_i} + \beta\frac{\partial KL}{\partial z_i}$$

$$\begin{aligned}\frac{\partial F}{\partial z_i} = \alpha\sum_j \left[\frac{2ab\left\|z_i - z_j\right\|_2^{2(b-1)} p_{ij}}{1 + a\left\|z_i - z_j\right\|_2^{2b}} - \frac{2b(1 - p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1 + a\left\|z_i - z_j\right\|_2^{2b})}\right]\left\|z_i - z_j\right\| \\ +\beta\sum_k \frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}}\end{aligned} \tag{C.6}$$

Then the update of z_i is performed using $\frac{\partial F}{\partial z_i}$, we just have to choose α and β.

C.3 The derivative of Objective Function w.r.t. the centroids

Now, we derivate our objective function w.r.t. μ_k. First, let's recall our weighted sum function:

$$F(P,Q,T,S) = \alpha CE(P,Q) + \beta KL(T||S) \tag{C.7}$$

Such that:

$$CE(P,Q) = \sum_i \sum_j \left[p_{ij} \log(\frac{p_{ij}}{q_{ij}}) + (1-p_{ij}) \log(\frac{1-p_{ij}}{1-q_{ij}}) \right]$$

$$KL(T||S) = \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik}$$

Also, let's recall the derivative of our weighted sum function:

$$\frac{\partial F}{\partial \mu_k} = \alpha \frac{\partial CE}{\partial \mu_k} + \beta \frac{\partial KL}{\partial \mu_k}$$

CE doesn't depend on cluster centroids, so its derivative is zero. Therefore, we compute only the derivative of the KL divergence w.r.t. μ_k:

$$KL(T,S) = \sum_i \sum_k T_{ik} \log T_{ik} - T_{ik} \log S_{ik}$$

T_{ik} is considered as a constant number, since T_{ik} doesn't depend on μ_k. Thus the derivative of $T_{ik} \log T_{ik}$ w.r.t. μ_k is zero. The partial derivative of the KL function w.r.t. μ_k is:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i -\frac{\partial S_{ik}}{\partial \mu_k} \frac{T_{ik}}{S_{ik}} \tag{C.8}$$

To simplify the derivative of the KL divergence w.r.t. μ_k we calculate the derivative of S_{ik} w.r.t μ_k:

$$\frac{\partial S_{ik}}{\partial \mu_k} = \frac{2ab\|z_i - \mu_k\|_2^{2b-1}}{(1 + a\|z_i - \mu_k\|_2^{2b})^2} \tag{C.9}$$

Substituting S_{ik} and its derivative $\frac{\partial S_{ik}}{\partial \mu_k}$ (Eq. C.9) in Eq. C.8. we obtain:

$$\frac{\partial KL}{\partial \mu_k} = \sum_i -\frac{2ab\|z_i - \mu_k\|_2^{2b-1} T_{ik}}{1 + a\|z_i - \mu_k\|_2^{2b}} \tag{C.10}$$

Appendix D

Definition of the values domain of the CE and KL divergence gradients w.r.t. datapoints

D.1 Defining the interval values of the CE gradient w.r.t. datapoints

We have the gradient of the CE as follows:

$$\frac{\partial CE}{\partial z_i} = \sum_j \left[\frac{2bp_{ij}}{1/(a\left\|z_i - z_j\right\|_2^{2(b-1)}) + \left\|z_i - z_j\right\|_2^2} - \frac{2b(1-p_{ij})}{\left\|z_i - z_j\right\|_2^2 (1 + a\left\|z_i - z_j\right\|_2^{2b})} \right] \left\|z_i - z_j\right\| \quad \text{(D.1)}$$

We observe that the gradient of the CE has two terms $\frac{2bp_{ij}}{1/(a\left\|z_i-z_j\right\|_2^{2(b-1)})+\left\|z_i-z_j\right\|_2^2}$ and $\frac{2b(1-p_{ij})}{\left\|z_i-z_j\right\|_2^2(1+a\left\|z_i-z_j\right\|_2^{2b})}$. So we need to study their values Domain. Let us start with the first term:

$$\frac{2bp_{ij}}{1/(a\left\|z_i - z_j\right\|_2^{2(b-1)}) + \left\|z_i - z_j\right\|_2^2} \quad \text{(D.2)}$$

We have the values domain of P and Q are $[0,1]$. In addition, the hyper-parameters a

and b are always positives. Also, we have $\left\|z_i - z_j\right\|_2^{2(b-1)}$ and $\left\|z_i - z_j\right\|_2^2$ are always positives, so the values domain of the first term (Eq. D.2) is $[0, +\infty[$. Now let us move to the second term:

$$-\frac{2b(1-p_{ij})}{\left\|z_i - z_j\right\|_2^2(1+a\left\|z_i - z_j\right\|_2^{2b})} \tag{D.3}$$

In the above term (Eq. D.3), we observe that the negative sign precedes it, so the values domain of this term is $]-\infty, 0]$. So the values domain of the CE gradient is $]-\infty, +\infty[$.

D.2 Defining the values domain of the KL divergence gradient w.r.t. datapoints

In this part, we studied the definition domain of the KL divergence gradient w.r.t z_i in the two variants.

D.2.1 The derivative of the KL divergence Plus variant

Let us recall the derivative of the KL divergence w.r.t. z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \left[\frac{\partial T_{ik}}{\partial z_i}(1+\log\frac{T_{ik}}{S_{ik}}) + \frac{2bT_{ik}}{1/\|z_i-\mu_k\|_2^{2b-1} + \|z_i-\mu_k\|_2^2}\right] + \sum_{i'\neq i}\sum_k \left[\frac{\partial T_{i'k}}{\partial z_i}(1+\log\frac{T_{i'k}}{S_{i'k}})\right]$$

The same thing for the KL gradient. We proved that the values domain of S_{ik} and T_{ik} are in $[0, 1]$ in the division by zero subsection (see Appendix B). So the values domain of KL derivative in $[0, +\infty[$.

D.2.2 The derivative of the KL divergence Light Plus variant

Let us recall the derivative of the KL divergence w.r.t. z_i:

$$\frac{\partial KL}{\partial z_i} = \sum_k \frac{2ab\|z_i-\mu_k\|_2^{2b-1}T_{ik}}{1+a\|z_i-\mu_k\|_2^{2b}}$$

We have the values domain of T_{ik} is in $[0, 1]$, $\left\|z_i - z_j\right\|_2^2$ is always positive, and a is also positive, so the values domain of KL derivative in $[0, +\infty[$.

Bibliography

[1] C. Bishop, "Pattern recognition and machine learning," *Springer google schola*, vol. 2, pp. 5–43, 2006.

[2] M. Espadoto, R. M. Martins, A. Kerren, N. S. Hirata, and A. C. Telea, "Toward a quantitative survey of dimension reduction techniques," *IEEE transactions on visualization and computer graphics*, vol. 27, no. 3, pp. 2153–2173, 2019.

[3] J. B. Tenenbaum, V. d. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[4] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[5] J. Miao and L. Niu, "A survey on feature selection," *Procedia Computer Science*, vol. 91, pp. 919–926, 2016.

[6] J. MacQueen *et al.*, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, no. 14. Oakland, CA, USA, 1967, pp. 281–297.

[7] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*. PMLR, 2016, pp. 478–487.

[8] K. Ghasedi Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization,"

in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5736–5745.

[9] J. Chang, L. Wang, G. Meng, S. Xiang, and C. Pan, "Deep adaptive image clustering," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 5879–5887.

[10] Y. Yan, H. Hao, B. Xu, J. Zhao, and F. Shen, "Image clustering via deep embedded dimensionality reduction and probability-based triplet loss," *IEEE Transactions on Image Processing*, vol. 29, pp. 5652–5661, 2020.

[11] M. L. Kherfi, "Review of human-computer interaction issues in image retrieval," *Advances in Human Computer Interaction. IntechOpen*, pp. 215–240, 2008.

[12] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[13] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, vol. 11, no. 12, 2010.

[14] X. Huang, L. Wu, and Y. Ye, "A review on dimensionality reduction techniques," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 33, no. 10, p. 1950017, 2019.

[15] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.

[16] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[17] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[18] M. Ranzato, F. J. Huang, Y.-L. Boureau, and Y. LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 2007, pp. 1–8.

[19] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[20] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[24] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[27] D. Xu and Y. Tian, "A comprehensive survey of clustering algorithms," *Annals of Data Science*, vol. 2, no. 2, pp. 165–193, 2015.

[28] J. Wang, J. Wang, Q. Ke, G. Zeng, and S. Li, "Fast approximate k-means via cluster closures," in *Multimedia data mining and analytics*. Springer, 2015, pp. 373–395.

[29] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[30] K. C. Gowda and G. Krishna, "Agglomerative clustering using the concept of mutual nearest neighbourhood," *Pattern recognition*, vol. 10, no. 2, pp. 105–112, 1978.

[31] W.-T. Wang, Y.-L. Wu, C.-Y. Tang, and M.-K. Hor, "Adaptive density-based spatial clustering of applications with noise (dbscan) according to data," in *2015 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 1. IEEE, 2015, pp. 445–451.

[32] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 160–172.

[33] A. Y. Ng, M. I. Jordan, Y. Weiss *et al.*, "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[34] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[35] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004, pp. 551–556.

[36] E. Elhamifar and R. Vidal, "Sparse manifold clustering and embedding," *Advances in neural information processing systems*, vol. 24, 2011.

[37] V. M. Patel and R. Vidal, "Kernel sparse subspace clustering," in *2014 ieee international conference on image processing (icip)*. IEEE, 2014, pp. 2849–2853.

[38] U. Shaham, K. Stanton, H. Li, B. Nadler, R. Basri, and Y. Kluger, "Spectralnet: Spectral clustering using deep neural networks," *arXiv preprint arXiv:1801.01587*, 2018.

[39] N. Mrabah, N. M. Khan, R. Ksantini, and Z. Lachiri, "Deep clustering with a dynamic autoencoder: From reconstruction towards centroids construction," *Neural Networks*, vol. 130, pp. 206–228, 2020.

[40] E. Min, X. Guo, Q. Liu, G. Zhang, J. Cui, and J. Long, "A survey of clustering with deep learning: From the perspective of network architecture," *IEEE Access*, vol. 6, pp. 39 501–39 514, 2018.

[41] E. Aljalbout, V. Golkov, Y. Siddiqui, M. Strobel, and D. Cremers, "Clustering with deep learning: Taxonomy and new methods," *arXiv preprint arXiv:1801.07648*, 2018.

[42] J. Guo, X. Yuan, P. Xu, H. Bai, and B. Liu, "Improved image clustering with deep semantic embedding," *Pattern Recognition Letters*, vol. 130, pp. 225–233, 2020.

[43] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[44] J. Yang, D. Parikh, and D. Batra, "Joint unsupervised learning of deep representations and image clusters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5147–5156.

[45] W. Hu, T. Miyato, S. Tokui, E. Matsumoto, and M. Sugiyama, "Learning discrete representations via information maximizing self-augmented training," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1558–1567.

[46] B. S. Anderson, "Using text mining to glean insights from covid-19 literature," *Journal of Information Science*, p. 01655515211001661, 2021.

[47] E. M. Eren, N. Solovyev, and E. Nicholas, Charles.and Raff, "Covid-19 literature clustering," April 2020, malware Research Group.

[48] S. Reddy, B. Ravi, P. Sandosh, V. Karin, M. Chaitanya, L. Rani, M. Ville-Petteri, P. Smitan, K. Puru, and S. Saumya, "Use and validation of text mining and cluster algorithms to derive insights from corona virus disease-2019 (covid-19) medical literature," *Computer Methods and Programs in Biomedicine*, vol. 1, p. 100010, 2021.

[49] M. Allaoui, M. L. Kherfi, and A. Cheriet, "Considerably improving clustering algorithms using umap dimensionality reduction technique: A comparative study," in *International Conference on Image and Signal Processing*. Springer, 2020, pp. 317–325.

[50] M. Allaoui, M. L. Kherfi, A. Cheriet, and A. Bouchachia, "Unified embedding and clustering," *TechRxiv, Preprint*, 2021. [Online]. Available: https://doi.org/10.36227/techrxiv.16926754.v1

[51] M. Allaoui, N. E.-H. S. B. Aissa, A. B. Belghith, and M. L. Kherfi, "A machine learning-based tool for exploring covid-19 scientific literature," in *2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI)*. IEEE, 2021, pp. 1–7.

[52] A. Ultsch, "U* c: Self-organized clustering with emergent feature maps." in *LWA*. Citeseer, 2005, pp. 240–244.

[53] P. M. Baggenstoss, "Statistical modeling using gaussian mixtures and hmms with matlab," *Naval Undersea Warfare Center, Newport RI*, 2002.

[54] J. J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 16, no. 5, pp. 550–554, 1994.

[55] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[56] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[57] D. D. Lewis, Y. Yang, T. Russell-Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *Journal of machine learning research*, vol. 5, no. Apr, pp. 361–397, 2004.

[58] L. L. Wang, K. Lo, Y. Chandrasekhar, R. Reas, J. Yang, D. Eide, K. Funk, R. Kinney, Z. Liu, W. Merrill *et al.*, "Cord-19: The covid-19 open research dataset," *ArXiv*, 2020.

[59] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[60] T. Calinski and J. Harabasz, "A dendrite method for cluster analysis," *Communications in Statistics-theory and Methods*, vol. 3, no. 1, pp. 1–27, 1974.

[61] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[62] D. L. Davies and D. W. Bouldin, "A cluster separation measure," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.

[63] ——, "A cluster separation measure," *IEEE transactions on pattern analysis and machine intelligence*, no. 2, pp. 224–227, 1979.

[64] B. Desgraupes, "Clustering indices," *University of Paris Ouest-Lab Modal'X*, vol. 1, p. 34, 2013.

[65] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

[66] R. McConville, R. Santos-Rodriguez, R. J. Piechocki, and I. Craddock, "N2d:(not too) deep clustering via clustering the local manifold of an autoencoded embedding," *arXiv preprint arXiv:1908.05968*, 2019.

[67] E. Becht, L. McInnes, J. Healy, C.-A. Dutertre, I. W. Kwok, L. G. Ng, F. Ginhoux, and E. W. Newell, "Dimensionality reduction for visualizing single-cell data using umap," *Nature biotechnology*, vol. 37, no. 1, pp. 38–44, 2019.

[68] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[69] E. Alpaydin and F. Alimoglu, "Pen-based recognition of handwritten digits data set. university of california, irvine," *Machine Learning Repository. Irvine: University of California*, vol. 4, no. 2, 1998.

[70] D. B. Graham and N. M. Allinson, "Characterising virtual eigensignatures for general purpose face recognition," in *Face Recognition*. Springer, 1998, pp. 446–456.

[71] W. Dong, C. Moses, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures," in *Proceedings of the 20th international conference on World wide web*, 2011, pp. 577–586.

[72] A. Dalmia and S. Sia, "Clustering with umap: Why and how connectivity matters," *arXiv preprint arXiv:2108.05525*, 2021.

[73] K. Ozaki, M. Shimbo, M. Komachi, and Y. Matsumoto, "Using the mutual k-nearest neighbor graphs for semi-supervised classification on natural language data," in *Proceedings of the fifteenth conference on computational natural language learning*, 2011, pp. 154–162.

[74] B. Luo, R. C. Wilson, and E. R. Hancock, "Spectral embedding of graphs," *Pattern recognition*, vol. 36, no. 10, pp. 2213–2230, 2003.

[75] M. C. Thrun and A. Ultsch, "Clustering benchmark datasets exploiting the fundamental clustering problems," *Data in brief*, vol. 30, p. 105501, 2020.

[76] M. C. Thrun, *Projection-based clustering through self-organization and swarm intelligence: combining cluster analysis with the visualization of high-dimensional data*. Springer, 2018.

[77] C. R. Center, "Covid-19 dashboard by the center for systems science and engineering (csse) at johns hopkins university (jhu)," *John Hopkins University & Medicine*, 2020.

[78] *Pneumonia of unknown cause – China*, 2019 (accessed September 2, 2020), https://www.who.int/csr/don/05-january-2020-pneumonia-of-unkown-cause-china/en/.

[79] M. S. Diamond and T. C. Pierson, "The challenges of vaccine development against a new virus during a pandemic," *Cell Host & Microbe*, vol. 27, no. 5, pp. 699–703, 2020.

[80] M. Nicola, Z. Alsafi, C. Sohrabi, A. Kerwan, A. Al-Jabir, C. Iosifidis, M. Agha, and R. Agha, "The socio-economic implications of the coronavirus pandemic (covid-19): A review," *International journal of surgery (London, England)*, vol. 78, p. 185, 2020.

[81] J. J. Palop, L. Mucke, and E. D. Roberson, "Quantifying biomarkers of cognitive dysfunction and neuronal network hyperexcitability in mouse models of alzheimer's disease: depletion of calcium-dependent proteins and inhibitory hippocampal remodeling," in *Alzheimer's Disease and Frontotemporal Dementia*. Springer, 2010, pp. 245–262.

[82] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[83] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[84] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Icml*, 2010.

[85] S.-i. Amari, "Backpropagation and stochastic gradient descent method," *Neurocomputing*, vol. 5, no. 4-5, pp. 185–196, 1993.

[86] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[87] C.-L. Liu, K. Nakashima, H. Sako, and H. Fujisawa, "Handwritten digit recognition: benchmarking of state-of-the-art techniques," *Pattern recognition*, vol. 36, no. 10, pp. 2271–2285, 2003.

[88] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: an efficient data clustering method for very large databases," *ACM sigmod record*, vol. 25, no. 2, pp. 103–114, 1996.

[89] K. Al Sharou, Z. Li, and L. Specia, "Towards a better understanding of noise in natural language processing," in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, 2021, pp. 53–62.

[90] O. A. Ismaili, V. Lemaire, and A. Cornuéjols, "A supervised methodology to measure the variables contribution to a clustering," in *Neural Information Processing: 21st International Conference, ICONIP 2014, Kuching, Malaysia, November 3-6, 2014. Proceedings, Part I 21*. Springer, 2014, pp. 159–166.