MixKMeans: Clustering Question-Answer Archives

Deepak P
Centre for Data Sciences and Scalable Computing
Queen's University Belfast, UK
deepaksp@acm.org

Abstract

Community-driven Question Answering (CQA) systems that crowdsource experiential information in the form of questions and answers and have accumulated valuable reusable knowledge. Clustering of QA datasets from CQA systems provides a means of organizing the content to ease tasks such as manual curation and tagging. In this paper, we present a clustering method that exploits the two-part question-answer structure in QA datasets to improve clustering quality. Our method, *MixKMeans*, composes question and answer space similarities in a way that the space on which the match is higher is allowed to dominate. This construction is motivated by our observation that semantic similarity between question-answer data (QAs) could get localized in either space. We empirically evaluate our method on a variety of real-world labeled datasets. Our results indicate that our method significantly outperforms state-of-the-art clustering methods for the task of clustering question-answer archives.

1 Introduction

Community-based Question Answering (CQA) systems such as Yahoo! Answers[1], StackOverflow[2] and Baidu Zhidao[3] have become dependable sources of knowledge to solve common user problems. Unlike factoid question answering[4], CQA systems focus on crowdsourcing *how* and *why* questions and their answers. As is the case with any system where content is generated by web users, the generated content would be of varying quality, reliability, readability and abstraction. Thus, manual curation of such datasets is inevitable to weed out low quality and duplicate content to ensure user satisfaction. A natural way to aid manual curation of such broad-based CQA archives is to employ clustering so that semantically related QAs are grouped together; this would help organize the corpus in a way that experts engaged in manual curation be assigned specific clusters relating to areas of their expertise. Clustering also provides a platform to enable tagging the QA dataset; cluster topics could be used as tags, or other QAs in the same cluster could be tagged as being *related* to a QA. The fundamental difference between CQA archives and general text document collections is the existence of a two-part structure in QAs and the difference in lexical "character" between the question and answer parts. This *lexical chasm (i.e., gap)* (Berger et al., 2000) between question and answer parts has been a subject of much study, especially, in the context of improving QA retrieval. In this paper, we consider using the two-part structure in QAs for clustering CQA datasets.

Motivating Example: Table 1 lists four example QAs from the context of a CQA system focused on addressing myriad technical issues. These QAs have been tagged in the table with a manually identified root-cause to aid understanding; the root-cause is not part of the CQA data per se. $QA1$ and $QA2$ are seen to address related issues pertaining to routers, whereas $QA3$ and $QA4$ are focused on the same nar-

[1] http://answers.yahoo.com
[2] http://www.stackoverflow.com
[3] http://en.wikipedia.org/en/Baidu_Knows
[4] e.g., http://trec.nist.gov/data/qa.html

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 1576–1585,
Austin, Texas, November 1-5, 2016.

row issue dealing with java libraries. Since $QA1$ and $QA2$ address different problems, they may not be expected to be part of the same cluster in fine-grained clusterings. On the other hand, the solutions suggested in $QA3$ and $QA4$ are distinct and different legitimate solutions to the *same* problem cause. Thus, from a semantics perspective, it is intuitive that $QA3$ and $QA4$ should be part of the same cluster in any clustering of the CQA dataset to aid actioning on them together; a human expert might decide to merge the question parts and tag one of the answers as an *alternative* answer. Let us now examine the lexical relatedness between the pairs as illustrated in Table 2. State-of-the-art text similarity measures that quantify word overlaps are likely to judge $QA1$ and $QA2$ to be having a *medium* similarity when either the question-part or the answer-part are considered. For the pair $(QA3, QA4)$, the question-part similarity would be judged to be *high* and the answer-part similarity as *low*. Thus, the high similarity between the root-causes of $QA3$ and $QA4$ manifest primarily in their question-parts. Analogously, we observed that some QAs involving the same root-cause lead to high answer-part similarity despite poor question-part similarity. This is especially true in cases involving suggestion of the same sequence of solution steps despite the question-part being divergent due to focusing on different symptoms of the same complex problem. From these observations, we posit that high similarities on *either* the question-space or answer-space is indicative of semantic relatedness. Any clustering method that uses a sum, average or weighted sum aggregation function to arrive at pair-wise similarities, such as a K-Means clustering that treats the collated QA as a single document, would intuitively be unable to heed to such differential manifestation of semantic similarities across the two parts.

Our Contributions: We address the problem of harnessing the two-part structure in QA pairs to improve clustering of CQA data. Based on our observations on CQA data such as those illustrated in the example, we propose a clustering approach, *MixK-Means*, that composes similarities (dissimilarities) in the question and answer spaces using a max (min) operator style aggregation. Through abundant empirical analysis on real-world CQA data, we illustrate that our method outperforms the state-of-the-art approaches for the task of CQA clustering.

2 Related Work

To enable position our work in the context of existing literature, we now summarize prior work along three related directions, viz., (1) processing of CQA datasets, (2) multi-modal data clustering, and (3) K-Means extensions.

Processing CQA Datasets: Most work on processing CQA data has been in the realm of retrieval, where the task addressed is to leverage CQA datasets to aid answering new questions. These start with a new question and find one of (i) related questions (Zhou et al., 2015), (ii) potentially usable answers (Shtok et al., 2012), or (iii) related QAs (Xue et al., 2008). Different methods differ in the technique employed to overcome the *lexical chasm*, with statistical translation models (Brown et al., 1993) that model word-level correlations between questions and answers being the most popular tool for the same. Usage of topic models (e.g., (Cai et al., 2011)) and combining evidence from topic and translation models (Zhou et al., 2015) have also met with success. The usage of deep-learning methods such as deep belief networks (Wang et al., 2011) and auto-encoders (Zhou et al., 2016) have also been explored for QA retrieval. While the problem of estimating the relevance of a QA to address a new question is related to the problem of estimating similarities between QAs to aid clustering, the latter problem is different in that both question and answer parts are available at either side. In fact, our problem, CQA clustering, has been largely unexplored among literature in CQA data processing. In the interest of benchmarking our work against techniques from the CQA processing community, we consider the correlated latent representation learnt by the recent auto-encoder based neural network (AENN) method (Zhou et al., 2016) as input to K-Means, and empirically validate our technique against the AENN+K-Means combination (referred to as AENN, for short) in our experimental study. Outside the task of retrieval, there has been work on getting QAs from experience reports (Deepak et al., 2012) and discussion forums (P and Visweswariah, 2014). Conversational transcripts from contact centres, as outlined in (Kummamuru et al., 2009), form

#	QA	Cause
QA1	**Q:** My internet connection is not working, my router shows the "Internet" led blinking in red.	Router Authentication Issue
	A: Please go to the router login page and re-login with broadband credentials; click "connect" and you should be on the internet.	
QA2	**Q:** My internet connection is not working, only the power led is lit in the router.	Router Loose Connection
	A: Check whether the router login page is loading. Else, the broadband cable may not be connected properly.	
QA3	**Q:** My Java app is picking up the old dojo 0.4.4 libraries though I have a newer version.	Multiple Libraries in Classpath
	A: Search for dojo 0.4.4 in Windows, and delete off the folder, and it should automatically start using the newer version.	
QA4	**Q:** My java application is not picking up the new dojo 1.11.1 libraries that I just installed.	Multiple Libraries in Classpath
	A: Update the java classpath variable to exclude the path to the earlier version, and add the path to the new version.	

Table 1: Example CQA Data

QA Pair		Part	Lexical Similarity
QA1	QA2	Question	Medium
QA1	QA2	Answer	Medium
QA3	QA4	Question	High
QA3	QA4	Answer	Low

Table 2: Similarity Analysis of QAs from Table 1

another rich source of QA data, but need careful segmentation due to interleaving of question and answer parts.

Multi-modal Data Clustering: The problem of clustering CQA data is an instance of the general problem of clustering multi-modal (aka multi-relational, multi-view or heterogeneous) data when the question and answer parts are seen as instantiations of the same root cause, but in question and answer 'modalities'. Clustering multi-modal data has been explored well in the context of multi-media data clustering where each data element comes in multi-modal form such as *[image, caption]* pairs or *[audio, text]* pairs. The pioneering work in this field adapted markov random fields (Bekkerman and Jeon, 2007) to generate separate clusterings for each modality. Later approaches are closer to our task of generating a unified clustering across modalities; they work by learning a unified latent space embedding of the dataset, followed by usage of K-Means clustering (MacQueen and others, 1967). Eigen-decomposition (Petkos et al., 2012), spectral methods (Blaschko and Lampert, 2008) and canonical correlation analysis (Jin et al., 2015) have been exploited for learning the latent space prior to the clustering step. A recent work (Meng et al., 2014) proposes a single-pass leader-clustering[5] style formulation called GHF-ART to progressively assign data objects to clusters. Unlike most other methods that assume that vector representations are obtained from general multimedia data, the authors of GHF-ART lay out how text data be pre-processed for usage in GHF-ART, making it an appropriate method for usage in our setting. Accordingly, we will use GHF-ART as a baseline method for our experimental study.

K-Means Extensions: The method that we propose in this paper, *MixKMeans*, draws generous inspiration from the classical K-Means clustering formulation (MacQueen and others, 1967). There have been numerous extensions to the basic K-Means formulation over the last many decades; many such extensions have been covered in focused surveys (Steinley, 2006; Jain, 2010). Of particular interest in our scenario are those relating to usage of varying (dis)similarity measures. (Patel and Mehta, 2012) discuss the usage of various popular distance measures within the K-Means framework. The point-symmetry distance, where the distance between an object and the cluster prototype is determined using other objects' information, has been explored (Su and Chou, 2001) for usage within K-Means for face recognition applications. Another work (Visalakshi and Suguna, 2009) suggests the computation of the aggregate distance as a fraction of the distance

[5] https://cran.r-project.org/web/packages/leaderCluster/index.html

along the closest attribute to that along the farthest attribute. Despite the plethora of work around extending K-Means to work with a variety of methods to aggregate distances across attributes, we have not come across previous work composing distances at the level of attribute sets (or modalities) like we will do in this work.

3 Problem Definition

Let $\mathcal{D} = \{(q_1, a_1), \ldots, (q_n, a_n)\}$ be a dataset of QAs from a CQA archive where each answer a_i was posted in response to the corresponding question q_i. The CQA clustering problem is the task of partitioning $\mathcal{D}$ into k clusters $\mathcal{C} = \{C_1, \ldots, C_k\}$ where $\cup_i C_i = \mathcal{D}$ and $\forall(i,j), i \neq j \Rightarrow C_i \cap C_j = \phi$ (disjointedness) hold such that similar QAs are grouped into the same cluster and dissimilar QAs are assigned to different clusters. The key aspect that differentiates the CQA clustering problem from general clustering of relational data is the opportunity to leverage the specifics of the CQA data, such as the two-part structure, to model the similarity measure that would drive the clustering.

Evaluation: The quality of a clustering method may be quantified by assessing how well the clustering it produces, i.e., $\mathcal{C}$, reflects the semantic similarities between QAs in $\mathcal{D}$. Given a QA $(q_i, a_i) \in \mathcal{D}$, the other QAs that share the same cluster may be thought of as the *result* set, i.e., the set of *related* QAs according to $\mathcal{C}$. In a labeled dataset such as CQADupStack (Hoogeveen et al., 2015) where related QA pairs have been manually identified for each (q_i, a_i), the quality of the results set may be assessed by contrasting against the labeled set using a standard metric such as F-score[6]. These QA-specific F-scores are then aggregated across the QAs in $\mathcal{D}$ to arrive at a single quality measure for the clustering. We will use such aggregated dataset-level F-scores as our primary evaluation measure. It may be noted that the *related* labellings may not be "clustering-friendly"; for example, there may not exist any k-clustering with no *related* labels going across clusters. Additionally, we observed that not all related QAs were labeled to be related in the CQADupStack dataset. The dataset owners confirm the problem of missing labelings in a very recent study (Hoogeveen et al., 2016). It is conceivable that only a few potential results were manually inspected to inject labellings. Thus, while the relative trends on F-score offer insights, the absolute F-scores may only be treated as a loose lower bounds.

[6] https://en.wikipedia.org/wiki/F1_score

4 *MixKMeans*: Our Method

We now describe the key details of our proposed technique, *MixKMeans*. The name is motivated by the flexibility that is built into the method to *mix* (dis)similarities across question and answer spaces in a formulation that derives inspiration from the classical K-Means algorithm (MacQueen and others, 1967). Throughout this formulation, we represent question and answer parts of QAs by their respective tf-idf vectors. We start with our objective function and move on to the iterative optimization.

4.1 Objective Function

Guided by our observation from Section 1 that the space in which a pair of QAs are more similar should hold sway in determining their overall match, we outline a penalty function for a clustering $\mathcal{C}$:

$$\mathcal{O}^* = \sum_{C \in \mathcal{C}} \sum_{(q,a) \in C} min\{w_q \, d(q, C.\mu.q), w_a \, d(a, C.\mu.a)\} \quad (1)$$

where $C.\mu = (C.\mu.q, C.\mu.a)$ is a prototypical QA vector for cluster C and the parameter pair $[w_q, w_a]$ control the relative weighting between question and answer parts. $d(.,.)$ is a dissimilarity function modeled as a simple sum of squares of element-wise differences between vector entries, i.e., $d(x,y) = \sum_i (x[i] - y[i])^2$.

Intuitively, $\mathcal{O}^*$ sums up the distance between each QA in $\mathcal{D}$ and the prototypical QA vector of the cluster to which it is assigned to, making it a penalty function. Since we use dissimilarities that are inversely related to similarities, the min function captures the idea that the aggregate (dis)similarity be estimated according to the measure in the best matching space. For optimization convenience, we replace the min construction by a differentiable approximation to get a modified objective function:

$$\mathcal{O} = \sum_{C \in \mathcal{C}} \sum_{(q,a) \in C} \Bigg(\Big(w_q d(q, C.\mu.q) \Big)^x + \Big(w_a d(a, C.\mu.a) \Big)^x \Bigg)^{\frac{1}{x}} \quad (2)$$

where x is a reasonably high negative value or $x \to -\infty$. This is used since $(a^x + b^x)^{1/x}$ approximates $min\{a, b\}$ for high negative values of x. It is worth noting that the opposite effect (i.e., max approximation) is achieved when $x \to \infty$ for usage in scenarios where a max combination is desirable. The remainder of the steps are applicable for positive values of x too.

4.2 Optimization Approach

There are two sets of variables in Equation 2, viz., cluster assignments of QAs in $\mathcal{D}$ and the cluster prototypes ($C.\mu$s). We optimize for each set of variables alternatively, much like in the EM-steps used in the classical K-Means algorithm.

4.2.1 Estimating Cluster Memberships

The cluster membership estimation directly falls out from the objective function and the current estimates of cluster prototypes since $\mathcal{O}$ (Equation 2) involves an instance-specific term for each QA. We will simply assign each QA to the cluster such that the respective instance-specific term is minimized:

$$Cluster((q,a)) = \arg\min_{C \in \mathcal{C}} \Big(d^x_{Q+A}((q,a), C.\mu) \Big)^{\frac{1}{x}} \quad (3)$$

$d^x_{Q+A}(.,.)$ is a short-hand for composite distance, composed of two terms (which we will denote as $d^x_Q(.,.)$ and $d^x_A(.,.)$ respectively):

$$d^x_{Q+A}((q,a), C.\mu^\circ) = \big(w_q \times d(q, C.\mu^\circ.q) \big)^x + \big(w_a \times d(a, C.\mu^\circ.a) \big)^x \quad (4)$$

4.2.2 Estimating Cluster Prototypes

We now estimate the cluster prototype in element-wise fashion. Consider a particular element in the $C.\mu.q$ vector, $C.\mu.q[i]$; computing the partial derivative and simplifying:

$$\frac{\partial \mathcal{O}}{\partial C.\mu.q[i]} = \sum_{(q,a) \in C} \Bigg[-2 \Big(d^x_{Q+A}((q,a), C.\mu) \Big)^{\frac{1-x}{x}} d^{x-1}_Q((q,a), C.\mu)\ w_q\ (q[i] - C.\mu.q[i]) \Bigg] \quad (5)$$

Equating the first derivative to zero and solving for $C.\mu.q[i]$ gets us to the following form:

$$C.\mu.q[i] = \frac{\sum_{(q,a) \in C} q[i] \Bigg[\Big(d^x_{Q+A}((q,a), C.\mu^\circ) \Big)^{\frac{1-x}{x}} d^{x-1}_Q((q,a), C.\mu^\circ) \Bigg]}{\sum_{(q,a) \in C} \Bigg[\Big(d^x_{Q+A}((q,a), C.\mu^\circ) \Big)^{\frac{1-x}{x}} d^{x-1}_Q((q,a), C.\mu^\circ) \Bigg]} \quad (6)$$

where $C.\mu^\circ$ is used to indicate the estimate of $C.\mu$ from the previous iteration. The corresponding estimation for $C.\mu.a[i]$ is:

$$C.\mu.a[i] = \frac{\sum_{(q,a) \in C} a[i] \Bigg[\Big(d^x_{Q+A}((q,a), C.\mu^\circ) \Big)^{\frac{1-x}{x}} d^{x-1}_A((q,a), C.\mu^\circ) \Bigg]}{\sum_{(q,a) \in C} \Bigg[\Big(d^x_{Q+A}((q,a), C.\mu^\circ) \Big)^{\frac{1-x}{x}} d^{x-1}_A((q,a), C.\mu^\circ) \Bigg]} \quad (7)$$

Equations 6 and 7 form the cluster prototype estimation steps of our method. It may be noted that for the choice of parameters ($x = 1, w_q = w_a$), either equations reduce it to the usual centroid estimation process for K-Means (since the terms within $[\ldots]$ reduce to 1.0), as intuitively expected. Thus, the modified formulation generalizes K-Means by allowing to weigh each element differently, the weight being modeled as a product two components:

- First component involves $d^x_{Q+A}(.,.)$ and is a function of the composite distance of (q,a) to the cluster prototype.
- Second component involves one of $d^{x-1}_Q(.,.)$ or $d^{x-1}_A(.,.)$ and is a function of the respective space (Q or A) to which the specific vector element belongs.

Alg. 1 ***MixKMeans***

Input. Dataset $\mathcal{D}$, number of clusters k
Hyper-parameters: x, w_q, w_a
Output. Clustering $\mathcal{C}$

1: *Initialize* $C.\mu$s *using data points from* $\mathcal{D}$
2: **while** *not yet converged* **do**
3: $\quad \forall (q,a) \in \mathcal{D}$, *assign cluster using Eq. 3*
4: $\quad \forall C \in \mathcal{C}$, *estimate* $C.\mu$ *using Eq. 6 & 7*
5: **end while**
6: *Return current clustering assignments as* $\mathcal{C}$

4.3 *MixKMeans*: The Algorithm

Having outlined the various steps, we are now ready to present the overall $MixKMeans$ algorithm in Algorithm 1. As the pseudo-code indicates, the cluster assignment and prototype estimation steps are run in a loop until the clustering converges. Additionally, we terminate after a threshold number of iterations even if the clustering does not converge by then; we set the threshold to 10.

Initialization: In the initialization step, we initialize the first cluster prototype using a random QA from $\mathcal{D}$. Each of the next cluster prototypes are initialized using the QA that has the highest sum of distances to all pre-chosen cluster prototypes, distance computed using $(d^x_{Q+A}(.,.))^{1/x}$. This is inspired by previous work on spreading out the cluster centroids (Arthur and Vassilvitskii, 2007) in K-Means initialization.

Hyperparameters: The algorithm has three hyper-parameters, viz., the exponentiation parameter x and the weight parameters w_q and w_a. As outlined in Sec. 4.1, x should be a negative value; we observed that any value beyond -3.0 does not make any significant differences to the final clustering (while higher absolute values for the exponent pose an underflow risk) and thus use $x = -3.0$ consistently. For the weights, we set $w_q = 0.2$ and $w_a = 0.8$. Due to the min-formulation in the objective function, a lower weight increases the influence of the respective space. Thus, we let our composed similarities be influenced more by the question-space similarities as in previous work (Xue et al., 2008).

4.4 Generalizing *MixKMeans*

Since the question and answer spaces are neatly segregated into different terms in the parameter update equations, ***MixKMeans*** is easily generalizable to work with more than two spaces. Consider the set of spaces to be $\mathcal{M} = \{\ldots, M, \ldots\}$ and that each object, $X \in \mathcal{D}$ be represented by an $|\mathcal{M}|$ tuple; now, the modified update equations are as follows:

$$Cluster(X) = \arg\min_{C \in \mathcal{C}} \left(d_{\sum_{M \in \mathcal{M}} M}(X, C.\mu) \right)^{\frac{1}{x}} \tag{8}$$

$$C.\mu.M[i] = \frac{\sum_{X \in C} X.M[i] \left[\left(d^x_{\sum_{M \in \mathcal{M}} M}(X, C.\mu^\circ) \right)^{\frac{1-x}{x}} d^{x-1}_M(X, C.\mu^\circ) \right]}{\sum_{X \in C} \left[\left(d^x_{\sum_{M \in \mathcal{M}} M}(X, C.\mu^\circ) \right)^{\frac{1-x}{x}} d^{x-1}_M(X, C.\mu^\circ) \right]} \tag{9}$$

where the somewhat awkward notation $d^x_{\sum_{M \in \mathcal{M}} M}(.,.)$ denotes the direct generalization of $d^x_{Q+A}(.,.)$ to cover all spaces in $\mathcal{M}$.

A simple modeling extension to use the generalized *MixKMeans* in the CQA setting is to consider the question title and question description as two separate spaces instead of using a single question space, increasing the total number of spaces to three; such a split of the question-part was used in (Qiu et al., 2013). In certain cases, one might want to use spaces that are of questionable quality due to reasons such as sparsity (e.g., set of tags associated with a question) and reliability (e.g., comments attached to a QA that could be noisy). The best way to leverage such spaces would be to include it in $\mathcal{M}$ for the modeling, but use a high weight for w_M; due to the min-style construction in the objective function, that setting will ensure that that space is called into play only when (a) signals from other spaces are not very strong, and (b) the signal from the space in question is very strong.

5 Experimental Evaluation

5.1 Datasets, Baselines and Setup

Datasets: We use the recently released data col-

lection, CQADupStack (Hoogeveen et al., 2015), for our experimental evaluation. Unlike most other datasets, this has each QA labeled with a set of *related* QAs, as alluded to in Section 3; this makes automated evaluation feasible in lieu of a laborious user study. We use the *android*, *gis*, *stats* and *physics* datasets from the CQADupStack collection, with our choice of datasets motivated by dataset size. These datasets comprise 2193, 3726, 4004 and 5044 QAs respectively.

Baselines: We use two baselines from literature in our study, (i) AENN (Zhou et al., 2016), (ii) GHF-ART (Meng et al., 2014). AENN, as alluded to in Section 2, refers to the K-Means clustering in the latent space learnt by correlated auto-encoders across the Q-A subspaces. AENN requires triplets of the form [question, answer, other answer] in the training phase; we populate the *other answer* part by the answer to a *related* question from the dataset (it may be noted that this is advantageous to AENN since it gets to *'see'* some *related* labelings in the training, whereas other methods can't). GHF-ART is the state-of-the-art multi-modal clustering approach that is targeted towards scenarios that involve a text modality. Unlike typical clustering algorithms that can generate a pre-specified (k) number of clusters, the number of clusters in the GHF-ART output is controlled by a *vigilance parameter*, ρ. Lower values of ρ result in smaller number of clusters and vice versa. A third intuitive baseline is the degenerate $x = 1$ instantiation of *MixKMeans*, which we will denote as $X1$. We are interested in evaluating the improvement achieved by *MixKMeans* over the best possible instantiation of $X1$; towards that, for every setting denoted by the combination $[dataset, k]$, we do a search over possible positive values of w_q and w_a within the locus of the line $w_q + w_a = 1$. It may be noted that this search space includes simple QA clustering using K-Means, being the case where $w_q = w_a = 0.5$. We collect the best result of $X1$ from across the grid-search for each setting. This approach, which we will denote as $X1^*$, while impractical in real scenarios due to usage of labeled data, gives an empirical upper bound of the accuracy of $X1$.

Experimental Setup: We use a latent space dimensionality of 2000 for AENN since we observed an accuracy peak around that value, and set GHF-ART parameters to their recommended values from the paper. For *MixKMeans*, we use tf-idf representation and set $(x = -3.0, w_q = 0.2, w_a = 0.8)$ as discussed earlier (Section 4.3). We use the F-score[7] measure to experimentally compare the approaches. The F-score is computed using the *related*

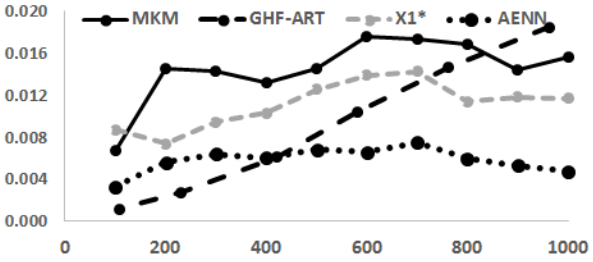

Figure 1: Android: F-Score (Y-Axis) vs. k

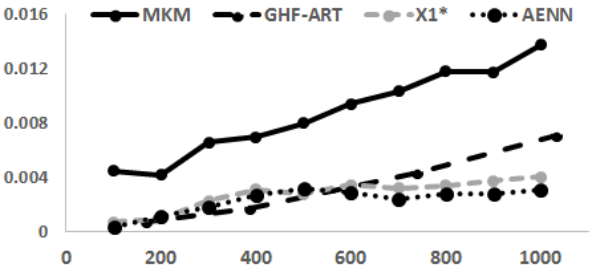

Figure 2: GIS: F-Score (Y-Axis) vs. k

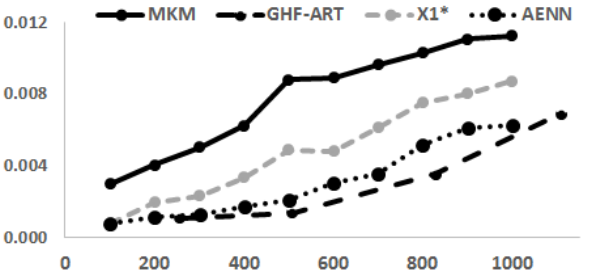

Figure 3: Stats: F-Score (Y-Axis) vs. k

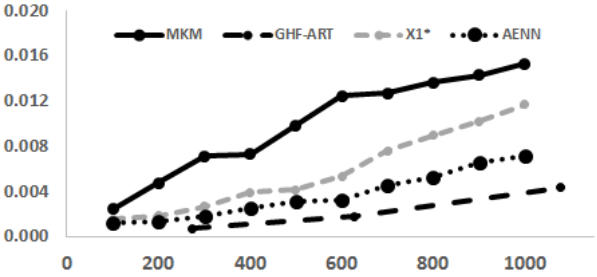

Figure 4: Physics: F-Score (Y-Axis) vs. k

[7] https://en.wikipedia.org/wiki/F1_score

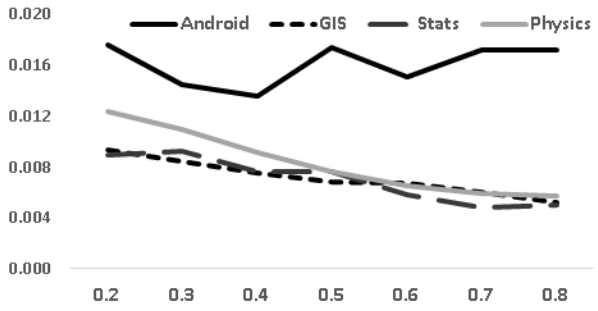

Figure 5: *MixKMeans*: F-Score (Y-Axis) vs. w_q at $k = 600$

labellings in the CQADupStack data, in a manner as described in Section 3. As pointed out therein, due to the sparse labellings, the F-score may only be regarded as a loose lower bound of their real values on a fully-labeled dataset.

5.2 Comparative Analysis

The results of the comparative analysis benchmarking our approach *MixKMeans* (MKM) against baselines $X1^*$, AENN and GHF-ART for the various datasets appear in Fig 1 (Android), Fig 2 (GIS), Fig 3 (Stats) and Fig 4 (Physics). Each of the trend-lines plot the F-Score against varying number of clusters in the output (k) in the range $\{100, 1000\}$. Since the number of clusters cannot be pre-specified for GHF-ART directly, we varied its ρ parameter to generate varying number of clusters to generate a trend-line that can be compared against *MixKMeans*, AENN and $X1^*$ directly. It may be noted that F-score is generally seen to increase when the clustering is more fine-grained (i.e., high k); this is an artifact of the sparse labeling that causes large clusters to have very low precision, causing precision and recall to diverge at low k, thus reducing the F-score. In most cases, *MixKMeans* is seen to outperform the other methods by scoring significantly higher in the F-Score, illustrating the superiority of our method. A notable exception appears in the higher values of k in the android dataset where GHF-ART quickly catches up and surpasses the others; however, it may be noted that $k \approx 1000$ is an extremely fine-grained clustering for the android dataset with 2193 QAs, and is thus not a very useful setting in practice. On the average, *MixKMeans* achieves an F-score improvement of between $30 - 100\%$ over the other methods.

5.3 *MixKMeans* Parameter Analysis

We now analyze the F-score trends of *MixKMeans* against varying values of the weight parameters. Since the relative weighting between w_q and w_a is what matters (simply scaling them both up by the same multiplier does not make any difference due to the construction of the objective), we set $w_a = (1.0 - w_q)$ and do the analysis for varying values of w_q keeping $k = 600$. As may be observed from the results in Figure 5, *MixKMeans* was seen to peak around $w_q = 0.2$-0.5 while degrading gracefully towards higher values of w_q. The android dataset, perhaps due to its relatively small size, records a different behavior as compared to the other trend-lines. Similar trends were observed for other values of k, indicating *MixKMeans* is not highly sensitive to the parameter and degrades gracefully outside the peak.

6 Conclusions and Future Work

We considered the problem of clustering question-answer archives from CQA systems. Clustering, we observed, helps in organizing CQA archival data for purposes such as manual curation and tagging. We motivated, by way of examples, as to why similarities along question and answer spaces be better composed using methods other than simple sum or average type aggregation. In particular, we noted that there are potentially different ways to answer questions pertaining to the same root-cause, mitigating the manifestation of the inherent root-cause similarity in the answer-space. Analogously, a sophisticated root-cause could be narrated differently by different people in the question part, while eliciting very similar answers. In short, we observe that legitimate reasons cause manifestation of semantic similarity between QAs to be localized on to one of the spaces. Accordingly, we propose a clustering method for QA archives, *MixKMeans*, that can heed to high similarities in either spaces to drive the clustering. *MixKMeans* works by iteratively optimizing the two sets of parameters, cluster assignments and cluster prototype learning, in an approach inspired by the classical K-Means algorithm. We empirically benchmark our method against current methods on multiple real-world datasets. Our experimental study illustrates that our method is able to significantly outperform other methods, establishing

MixKMeans as the preferred method for the task of clustering CQA datasets.

Future Work: As discussed in Section 4.4, *MixKMeans* is eminently generalizable to beyond two spaces. Considering the usage of other kinds of data (e.g., tags, comments) as additional "spaces" to extend the CQA clustering problem is an interesting direction for future work. The applicability of *MixKMeans* and it's *max* variant (i.e., with $x > 0$) for other kinds of multi-modal clustering problems from domains such as multimedia processing is worth exploring. The extension of the formulation to include a weight learning step may be appropriate for scenarios where prior information on the relative importance of the different spaces is not available. It is easy to observe that *MixKMeans* is prone to local optima issues; this makes devising better initialization strategies another potential direction. Yet another direction of interest is to make MixKMeans clusters interpretable, potentially by augmenting each cluster with word-level rules as used in earlier work on partitional document clustering (Balachandran et al., 2012).

References

David Arthur and Sergei Vassilvitskii. 2007. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics.

Vipin Balachandran, Deepak P, and Deepak Khemani. 2012. Interpretable and reconfigurable clustering of document datasets by deriving word-based rules. *Knowl. Inf. Syst.*, 32(3):475–503.

Ron Bekkerman and Jiwoon Jeon. 2007. Multi-modal clustering for multimedia collections. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. 2000. Bridging the lexical chasm: statistical approaches to answer-finding. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–199. ACM.

Matthew B Blaschko and Christoph H Lampert. 2008. Correlational spectral clustering. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. 2011. Learning the latent topics for question retrieval in community qa. In *IJCNLP*, volume 11, pages 273–281.

P. Deepak, Karthik Visweswariah, Nirmalie Wiratunga, and Sadiq Sani. 2012. Two-part segmentation of text documents. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, pages 793–802.

Doris Hoogeveen, Karin M Verspoor, and Timothy Baldwin. 2015. Cqadupstack: A benchmark data set for community question-answering research. In *Proceedings of the 20th Australasian Document Computing Symposium*, page 3. ACM.

Doris Hoogeveen, Karin M Verspoor, and Timothy Baldwin. 2016. Cqadupstack: Gold or silver?

Anil K Jain. 2010. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666.

Cheng Jin, Wenhui Mao, Ruiqi Zhang, Yuejie Zhang, and Xiangyang Xue. 2015. Cross-modal image clustering via canonical correlation analysis. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Krishna Kummamuru, Deepak Padmanabhan, Shourya Roy, and L Venkata Subramaniam. 2009. Unsupervised segmentation of conversational transcripts. *Statistical Analysis and Data Mining*, 2(4):231–245.

James MacQueen et al. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Lei Meng, Ah-Hwee Tan, and Dong Xu. 2014. Semi-supervised heterogeneous fusion for multimedia data co-clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 26(9):2293–2306.

Deepak P and Karthik Visweswariah. 2014. Unsupervised solution post identification from discussion forums. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 155–164.

Vaishali R. Patel and Rupa G. Mehta, 2012. *Proceedings of the International Conference on Soft Computing for Problem Solving (SocProS 2011) December 20-22, 2011: Volume 2*, chapter Data Clustering: Integrating Different Distance Measures with Modified k-Means Algorithm, pages 691–700. Springer India, India.

Georgios Petkos, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2012. Social event detection using multimodal clustering and integrating supervisory signals.

In *Proceedings of the 2nd ACM International Conference on Multimedia Retrieval*, page 23. ACM.

Xipeng Qiu, Le Tian, and Xuanjing Huang. 2013. Latent semantic tensor indexing for community-based question answering. In *ACL (2)*, pages 434–439.

Anna Shtok, Gideon Dror, Yoelle Maarek, and Idan Szpektor. 2012. Learning from the past: answering new questions with past answers. In *Proceedings of the 21st international conference on World Wide Web*, pages 759–768. ACM.

Douglas. Steinley. 2006. K-means clustering: A half-century synthesis. *British Journal of Mathematical and Statistical Psychology*, 59(1):1–34.

Mu-Chun Su and Chien-Hsing Chou. 2001. A modified version of the k-means algorithm with a distance based on cluster symmetry. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):674–680.

N Karthikeyani Visalakshi and J Suguna. 2009. K-means clustering using max-min distance measure. In *Fuzzy Information Processing Society, 2009. NAFIPS 2009. Annual Meeting of the North American*, pages 1–6. IEEE.

Baoxun Wang, Bingquan Liu, Xiaolong Wang, Chengjie Sun, and Deyuan Zhang. 2011. Deep learning approaches to semantic relevance modeling for chinese question-answer pairs. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(4):21.

Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 475–482. ACM.

Tom Chao Zhou, Michael Rung-Tsong Lyu, Irwin King, and Jie Lou. 2015. Learning to suggest questions in social media. *Knowledge and Information Systems*, 43(2):389–416.

Guangyou Zhou, Yin Zhou, Tingting He, and Wensheng Wu. 2016. Learning semantic representation with neural networks for community question answering retrieval. *Knowledge-Based Systems*, 93:75–83.