Normalization of Social Models in Social Argumentation Frameworks

1 Preliminaries

Definition 1 (Social argumentation frameworks)**.** *A* social argumentation framework *is a 3-tuple* $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$*, where*

- $\mathcal{A}$ *is the set of arguments,*
- $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ *is a binary attack relation between arguments,*
- $V : \mathcal{A} \cup \mathcal{R} \to \mathbb{N} \times \mathbb{N}$ *is a voting function keeping the crowd's pro and con votes for each argument and attack relation.*

Definition 2 (Vote Aggregation Function)**.** *Given a totally ordered set* L *with top and bottom elements* $\top$*,* $\bot$*, and a framework* $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$*,* τ *is any function such that* $\tau : (\mathcal{A} \cup \mathcal{R}) \times 2^{(\mathcal{A} \cup \mathcal{R})} \to L$*.*

Definition 3 (Semantic Framework)**.** *A semantic framework is a 6-tuple* $\mathcal{S} = \langle L, \curlywedge_1, \curlywedge_2, \curlyvee, \neg, \tau \rangle$ *where:*

- L *is a totally ordered set with top and bottom elements* $\top$*,* $\bot$*, containing all possible valuations of an argument.*
- $\curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}} : L \times L \to L$*, are two binary algebraic operations used to restrict strengths to given values.*
- $\curlyvee : L \times L \to L$*, is a binary algebraic operation on argument valuations used to combine or aggregate valuations and strengths.*
- $\neg : L \to L$ *is a unary algebraic operation for computing a restricting value corresponding to a given valuation or strength.*
- τ *is a vote aggregation function which, given the votes, determines the social support of an object within a set of objects.*

Notation 1. *Let* $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ *be a SAF,* $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ *a semantic framework. Then, let*

- $\mathcal{O} = \mathcal{A} \cup \mathcal{R}$ *be the set of objects, composed by the union of the sets of arguments and attack relations,*

- – $V^{+}(o) \triangleq x$ *denote the number of positive votes for object* o,
  – $V^{-}(o) \triangleq y$ *denote the number of negative votes for object* o,
  – $V^{t} : \mathcal{O} \to \mathbb{N}$ *be a function s.t.* $V^{t}(o) \triangleq x + y$,

  whenever $V(o) = (x, y)$,

- $V : 2^{\mathcal{O}} \to 2^{\mathbb{N}\times\mathbb{N}}$ *be a function s.t.* $V(\mathcal{O}^{'}) = \{V(o) \mid o \in \mathcal{O}^{'}\}$,
- $V^{t} : 2^{\mathcal{O}} \to 2^{\mathbb{N}\times\mathbb{N}}$ *be a function s.t.* $V^{t}(\mathcal{O}^{'}) = \{V^{t}(o) \mid o \in \mathcal{O}^{'}\}$,
- $max : 2^{\mathbb{N}} \to \mathbb{N}$ *be a function s.t. it returns the maximum value amongst the natural numbers from the non-empty multiset given as the input.*
- $min : 2^{\mathbb{N}} \to \mathbb{N}$ *be a function s.t. it returns the minimum value amongst the natural numbers from the non-empty multiset given as the input.*
- $\tau : 2^{\mathcal{O}} \to 2^{L}$ *be a function s.t.* $\tau(A) = \{\tau(a) | a \in A\}$.
- $\mathcal{R}^{-}(a) \triangleq \{a_i \in \mathcal{A} : (a_i, a) \in \mathcal{R}\}$ *be the set of direct attackers of an argument* $a \in \mathcal{A}$,
- $$\bigvee_{x\in X} x \triangleq (((x_1 \curlyvee x_2) \curlyvee ...) \curlyvee x_n)$$

  $X = \{x_1, x_2, ..., x_n\}$ *denote the aggregation of a multiset of elements of* L.

Definition 4 (Model)**.** *Let* $F = \langle \mathcal{A}, \mathcal{R}, V\rangle$ *be a social argumentation framework,* $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau\rangle$ *be a semantic framework. An* $\mathcal{S}$*-model of* F *is a total mapping* $M : \mathcal{A} \to L$ *such that for all* $a \in \mathcal{A}$,

$$M(a) = \tau(a, \mathcal{A}) \curlywedge_{\mathcal{A}} \neg \bigvee_{a_i \in \mathcal{R}^{-}(a)} (\tau((a_i, a), \mathcal{R}) \curlywedge_{\mathcal{R}} M(a_i))$$

Definition 5 (Enhanced Vote Aggregation)**.** *Given a SAF* $F = \langle \mathcal{A}, \mathcal{R}, V\rangle$ *and a semantic framework* $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau\rangle$*, enhanced vote aggregation function* $\tau_e : \mathcal{O} \times 2^{\mathcal{O}} \to [0, 1]$ *is the vote aggregation function such that*

$$\tau_e(o, \mathcal{O}) = \begin{cases} 0 & V(o) = (0, 0) \\ \frac{V^{+}(o)}{V^{t}(o) + \frac{1}{max(V^{t}(o\cup\mathcal{O}))}} & otherwise \end{cases}$$

Definition 6 (Enhanced Product Semantics)**.** *An enhanced product semantic framework is any* $\mathcal{S}_e^{\cdot} = \langle [0, 1], \tau_e, \curlywedge^{\cdot}, \curlywedge^{\cdot}, \curlyvee^{\cdot}, \neg\rangle$ *where 1)* $x_1 \curlywedge^{\cdot} x_2 = x_1 \cdot x_2$*, 2)* $x_1 \curlyvee^{\cdot} x_2 = x_1 + x_2 - x_1 \cdot x_2$*, 3)* $\neg x_1 = 1 - x_1$

Conjecture 1. *Let* $F = \langle \mathcal{A}, \mathcal{R}, V\rangle$ *be a social argumentation framework,* $\mathcal{S}_e^{\cdot} = \langle [0, 1], \tau_e, \curlywedge^{\cdot}, \curlywedge^{\cdot}, \curlyvee^{\cdot}, \neg\rangle$ *be a semantic framework. Then,* F *has one and only one* $\mathcal{S}_e^{\cdot} - model$.

Notation 2. *Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a social argumentation framework, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau_e \rangle$ be a semantic framework. Then, under the assumption that Conjecture 1 holds, let*

- $\mathcal{D} \triangleq \{M(a) | a \in \mathcal{A}\}$ *be the multiset of model evaluations of all arguments.*

2 Normalization of Model Evaluations

2.1 Problem domain

We have some dataset $\mathcal{D}$ containing the multiset of model evaluation values for all the arguments of a social abstract argumentation framework with product semantics. The values $\forall d \in \mathcal{D}$ lay between the interval $[0, 1] \in \mathbb{R}$.

Under the aforementioned semantics, the values in our candidate datasets tend to form big clusters, especially close to zero. The main goal of this study is adjusting the original distributions by some mapping $\sigma : \mathcal{D} \to [0, 1]$, in order to spread high-density areas (increasing the values in most cases), so that arguments with distinct values can be distinguished better.

The method of remapping may introduce some distortions or biases into the data. However in this case distortions are deliberately introduced to the system, in order to better expose the information content.

2.1.1 Characterization of Normalized Sets

–Whole section Under Revision–

In this section we try to find out the parameters that define whether a dataset is *normalized* or not. However the findings are at a preliminary level.

At this point, two values that I believe to be relevant are:

- The ratio between the range of the model evaluations and the range of the social support values

$$\frac{max(\mathcal{D}) - min(\mathcal{D})}{max(\mathcal{T}) - min(\mathcal{T})}$$

- The ratio between the standard deviation of the model evaluations and the standard deviation of the social support values

$$\frac{\sigma(\mathcal{D})}{\sigma(\mathcal{T})}$$

In the following subsection you may find the narrative of the thought process behind determining up with these two elements. But before that, below with Definition 7 we take our first shot at defining what a normalized set is, by incorporating the aforementioned the parameters.

Definition 7 (Normalized dataset)**.** *Given a SAF* $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ *and a semantic framework* $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$*, the generic standard deviation mapping* $\sigma : 2^{\mathbb{N}} \to \mathbb{N}$*, the generic mean function* $\mu : 2^{\mathbb{N}} \to \mathbb{N}$ *and constants* $r_1, r_2 \in \mathbb{R}^+$*, the multiset of model evaluations* $\mathcal{D}$ *is said to be normalized dataset if the following conditions hold:*

- $r_1 - \frac{\mu(D)}{\mu(T)} \geq \dfrac{max(\mathcal{D}) - min(\mathcal{D})}{max(\mathcal{T}) - min(\mathcal{T})} \geq r_1 + \frac{\mu(D)}{\mu(T)}$,
- $r_2 - \frac{\sigma(D)}{\sigma(T)} \geq \dfrac{\sigma(\mathcal{D})}{\sigma(\mathcal{T})} \geq r_2 + \frac{\sigma(D)}{\sigma(T)}$.

2.1.2 Discussion on the Characterization of Normalized Sets

Our initial suspicion was that whenever all model evaluations are very close to zero, this problem originates because the argumentation graph is strongly connected and this collection of datapoints $\mathcal{D}$ needs normalization. However consider the following setting:

Example 1. *Let* $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ *be a SAF,* $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ *a semantic framework,* $\mathcal{A} = \{a_1, a_2, a_3\}$ *and* $\mathcal{R} = \emptyset$.

In addition assume that all the social support values are extremely low. Since the attack set is empty, the model evaluation values for the arguments will be equal their social support values. So here the structure of the graph is not to blame. The model evaluations are low because of the popular opinion and thus probably the dataset should be left this way, it should not be normalized.

In order to exaggerate the example and display the situation clearer, we have taken the attack set to be empty. One may argue that to identify such a scenario it's enough to check how dense the argumentation graph is. However even if $\mathcal{R}$ all possible edges at full strength, the resulting model evaluations would be pretty close to the previous values since the social support is low for all the arguments. *Thus the graph structure is not a reliable indicator in identifying whether a dataset that requires normalization.*

At this point I thought comparing the ranges of the social support values and the model evaluations might give us an idea. From Notation 1&2 please remember that $\mathcal{T}$ is the multiset of all social supports and $\mathcal{D}$ is the multiset of model evaluations of all arguments in the framework. In this particular example it would follow that $\frac{max(\mathcal{D}) - min(\mathcal{D})}{max(\mathcal{T}) - min(\mathcal{T})} \simeq 1$. This would hint that the model evaluations have not shifted too far from the social support values.

However it appears that considering the ranges by itself is not sufficient as well. To see this, consider the following example:

Example 2. *Let* $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ *be a SAF,* $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ *a semantic framework,* $\mathcal{A} = \{a_1, a_2, a_3\}$ *and* $\mathcal{R} = \{(a_1, a_2), (a_2, a_3), (a_3, a_1)\}$.

Assume all the arguments are at full strength. Then under product semantics the model evaluations would follow as $M(a_1) = M(a_2) = M(a_3) \simeq 0.33$.

Now let's add another argument to the system (also with perfect social support) that only attacks to the first argument:

Example 3. *Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a SAF, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ a semantic framework, $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$ and $\mathcal{R} = \{(a_1, a_2), (a_2, a_3), (a_3, a_1), (a_4, a_1)\}$.*

Now with the addition of a single argument and attack relation we have $M(a_4) = M(a_2) = 1$ and $M(a_1) = M(a_3) = 0$. The range of the social supports is the same, since all of them are at the same value. However the range of the model evaluations increased from a minimum of 0 to a maximum of 1. So on top of the comparison of ranges, we need some additional concept that would capture the essence of the distribution i.e. how datapoints are spread in the interval.

Since standard deviation measures how far a set of numbers is spread out, the relative measure of the two multisets' standard deviations, $\frac{\sigma(\mathcal{D})}{\sigma(\mathcal{T})}$ may overcome the shortcoming of only taking ranges into consideration.

With these two ratio values, we obtain information on both the relative size of the sub-interval the datapoints are located at and also on the pattern they are spread.

As you might see in Definition 7, we've tried to restrain the envisioned values of the two rations with some constants. Currently we've not settled on these values, and maybe in future we may choose to replace these constants with some other terms.

One thing that comes to mind is some sort of a supervized learning method. Perhaps we may assume a certain set of datasets as our training set. Then we may parameterize the constants with regards to our training set. That way we may fix the interval for the ratios of ranges and standard deviations.

Let us see one example on this notion to make it more clear. For simplicity, we will only consider the standard deviation of the model evaluations when we are talking about the characterization of normalized sets (i.e. instead of comparing the ratios of range and standard deviations of social support values and model evaluations as we originally intend, we just take into account the standard deviations of the model evaluations in this example).

Example 4. *Assume we have the following three datasets that are known to be normalized:*

- $D_1 = \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$
- $D_2 = \{0.05, 0.1, 0.3, 0.45, 0.75, 0.8, 0.9, 0.95\}$
- $D_3 = \{0.25, 0.35, 0.4, 0.5, 0.6, 0.7, 0.75\}$

And we would like to figure out whether the following two data sets are normalized or not with respect to the training set.

- $D_4 = \{0.05, 0.1, 0.1, 0.15, 0.2, 0.2, 0.85, 0.9, 0.95\}$
- $D_5 = \{0.25, 0.3, 0.4, 0.6, 0.75, 0.9, 0.9\}$

Then the standard deviations for the training set follow as: $\sigma_1 \cong 0.287, \sigma_2 \cong 0.337, \sigma_3 \cong 0.172$.

Thus we may conclude that for a dataset D_i*, the corresponding standard deviation value should fall into the interval* $0.172 \leq \sigma_i \leq 0.337$ *for* D_i *to be regarded as normalized.*

The standard deviations for the test set follow as: $\sigma_4 \cong 0.364, \sigma_5 \cong 0.254$.

Consequently we may determine that D_5 *is a normalized dataset, on the other hand* D_4 *is not.*

Undoubtedly we've to study this subject in more detail to find out the optimal range for these values. We might also uncover more parameters with respect to the notion of normalized sets.

2.2 Characterization of Normalizing Mappings

The approach we adopt in this section is first defining a list of desirable properties that a normalizing mapping $\sigma_X : X \to [0, 1]$ may possess, where X is a fixed multiset. Then in the next section we will follow by defining some concrete classes of normalizing mappings which contain a subset the properties defined in the current section.

Before we move on, here we may better take a small pause to discuss the structure of the normalizing mapping. The mapping of each datapoint to a value in the unit interval is carried out with respect to the whole dataset. So the information that the dataset if fixed should be included in the mapping symbol. Indeed, that's what the subscript in the function definition above stands for.

For example, assume two sets $A = \{1, 3, 5, 7\}$ and $B = \{5, 50, 500, 5000\}$. Undoubtedly it would follow as $\sigma_A(5) \neq \sigma_B(5)$.

For the upcoming property definitions, let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a SAF, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ a semantic framework, $M(a) = d \in \mathcal{D}$ be the model evaluation of an argument $a \in \mathcal{A}$, $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ a normalizing function with some metric $\bigtriangleup : L \times L \to \mathbb{R}$.

Property 1 (Bottom argument)**.** *The argument of a context with the bottom value does not attain any strength in the normalized mutliset.*

$$d = 0 \Rightarrow \sigma_{\mathcal{D}}(d) = 0$$

Property 2 (Guarantee for the existence of arguments)**.** *An argument with some social strength is never diminished to the value of zero through normalization.*

$$d \neq 0 \Rightarrow \sigma_{\mathcal{D}}(d) \neq 0$$

Property 3 (Non-decreasing affect of normalization)**.** *Normalized value of a model evaluation should be bigger or equal the original value (in order to prevent the values cramming into a small space close to zero).*

$$d \leq \sigma_{\mathcal{D}}(d)$$

Property 4 (Conservation of relative ordering)**.** *The relative order between the pairs of normalized model evaluations is preserved.*

$$d_1 \geq d_2 \implies \sigma_{\mathcal{D}}(d_1) \geq \sigma_{\mathcal{D}}(d_2)$$

Property 5 (Conservation of distance ordering)**.** *The relative order between the distance of pairs of normalized model evaluations is preserved.*

$$\bigtriangleup(d_1, d_2) \geq \bigtriangleup(d_3, d_4) \implies \bigtriangleup(\sigma_{\mathcal{D}}(d_1), \sigma_{\mathcal{D}}(d_2)) \geq \bigtriangleup(\sigma_{\mathcal{D}}(d_3), \sigma_{\mathcal{D}}(d_4))$$

Even all together, these properties still give way to very flexible mapping definitions. For instance the identity mapping $id : x \mapsto x$ would interestingly satisfy all of the aforementioned properties.

Thus we have to define some property based on a concept that could capture the essence of specifically dense areas of the set of model evaluations. In the light of this, we benefit from the study of cluster analysis.

Very crudely, our objective via utilizing cluster analysis is to identify groups of objects that are very similar with regard to their values. Before going into more detail, firstly we need to formally define what a *cluster* is. In the literature, there are many definitions with respect to clusters. In our context we will define them closely to the *partitions* from the classical set theory via updating the definition accordingly with respect to multisets.

Definition 8 (Cluster)**.** *A family of multisets $\mathcal{C}$ is a clustering of a multiset X and every element C of $\mathcal{C}$ is a cluster if and only if all of the following conditions hold:*

- *Set of clusters does not contain the empty set.*

$$\emptyset \notin \mathcal{C}$$

- *(Collectively exhaustive) The sum of the multisets in $\mathcal{C}$ is equal to X.*

$$\biguplus_{C \in \mathcal{C}} C = X$$

- *(Mutually exclusive) Distinct multisets do not contain shared elements.*

$$(C_i, C_j \in \mathcal{C}) \wedge (C_i \neq C_j) \implies (C_i \wedge C_j = \emptyset)$$

As mentioned clustering is the task of grouping a set of objects in such a way that objects in the same cluster are more similar (in some sense or another) to each other than to those in other cluster. So we should also define the concept of *similarity* in a formal manner. Different procedures adopt different metrics, $\bigtriangleup : S \times S \to \mathbb{R}$ where S is some set of valuations, when grouping the most similar objects into clusters. We have a single clustering variable, the model evaluation of arguments. Thus using the generic distance function as the similarity measure is a possibility.

We continue by stating two more preliminary definitions that we will utilize in our last property.

Definition 9 (Separated points)**.** *Let $\mathcal{C}$ be a clustering of some multiset X and $C_i, C_j \in \mathcal{C}$. Two distinct datapoints p, q are said to be separated if $p \in C_i$ and $q \in C_j$ when $i \neq j$.*

Notation 3. *Let Separated(p, q) be the shortcut of notation where p and q are two separated points from some clustering $\mathcal{C}$.*

Definition 10 (Spacing)**.** *Given a metric $\bigwedge$, the spacing of a clustering $\mathcal{C}$ is the minimum distance between any two separated points:*

$\bigwedge(X, Y) = min_{x \in X, y \in Y} \bigwedge(x, y)$ where X and Y are any two distinct clusters, and d(x,y) denotes the distance between the two elements x and y.

One last discussion before we continue to the last property is k, the number of clusters. Some clustering procedures require k to be defined by the user manually, and the rest compute it as the starting step of the procedure. The problem with the family of clustering methods that compute the number of clusters by themselves is that they don't scale well. They have a high time-complexity, in all cases exceeding the cubic time. Thus we may either use a method like Monte Carlo to generate an artificial k and clustering centers then continue with the efficient methods, or use the inefficient but *self-computing* methods initially and then switch to the efficient ones.

Property 6 (Max spacing over k-clustering)**.** *Given constant $k \in \mathbb{N}$, a normalizing function $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ maximizes the spacing over a clustering $\mathcal{C}$ with k-clusters.*

... Property 5 cannot be satisfied by any methodology in some cases (i.e. set of model evaluations include 0 and 1) together with some other properties. But with the introduction of the clustering related concepts, we define a relaxed version of the property by limiting the restriction within clusters.

Property 7 (Conservation of distance ordering within clusters)**.** *Given a clustering $\mathcal{C}$, the relative order between the distance of pairs of normalized model evaluations is preserved within each cluster $C \in \mathcal{C}$.*

$$\bigwedge(d_1, d_2) \geq \bigwedge(d_3, d_4) \implies \bigwedge(\sigma_{\mathcal{D}}(d_1), \sigma_{\mathcal{D}}(d_2)) \geq \bigwedge(\sigma_{\mathcal{D}}(d_3), \sigma_{\mathcal{D}}(d_4))$$
$$\text{where } d_1, d_2, d_3, d_4 \in C$$

After defining the *watered down version* of the property regarding the relative distance ordering, we may define the desired mapping which basically consists of the *non-conflicting* subset of the aforementioned properties.

Definition 11 (Desired mapping)**.** *Given a SAF $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$, a semantic framework $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$, the multiset of model evaluations of all arguments $\mathcal{D} = \{M(a) | a \in \mathcal{A}\}$ and some metric $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$, a normalizing mapping $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ with clustering $\mathcal{C}$ is said to be a desired mapping if it obeys the following properties:*

- *Bottom argument (Property 1),*

- *Guarantee for the existence of arguments (Property 2),*
- *Non-decreasing affect (Property 3),*
- *Conservation of the relative ordering (Property 4),*
- *Maximum spacing over clustering (Property 6),*
- *Conservation of the relative ordering within clusters (Property 7).*

2.3 Algorithms for Concrete Normalizing Mapping

[1]

...The premise here is that from Cantor's first uncountability proof, we know that the set of all real number is uncountably infinite. Thus for two arbitrary reel numbers, there is some other reel number whose value falls between those two. In the light of this...

2.3.1 Single-linked normalization

...Here we had the assumption that K is given as an input to the algorithm. Alternatively, it is possible to modify the algorithm to iterate till a certain threshold of average distance amongst all the clusters is reached.

Remark 1. *Crudely the normalizing algorithms consist of two parts. The first part uses some clustering method, in which we basically make a partition of the dataset and assign each point to some cluster. But keep in mind that this phase can be considered as a labeling phase, in the sense that the model evaluation values(points) are not modified.*

The following step is the update phase, where we normalize the values within clusters. So we may think of the normalizing method as the composition of two methods $\sigma = \gamma o \delta$, the labelling phase γ and the update phase δ.

Since the labelling phase does not modify the values, it seems as if we may focus on the mapping that the update phase implements for the proof of the desirable properties, except the max-spacing property which particularly focuses on clustering.

The update phase of the algorithm which contain the normalization routine within clusters is between lines 20 and 28. The objective function can be simply represented as:

$$\delta_{\mathcal{D}}(x) = \frac{x - min(C_i)}{max(C_i) - min(C_i)}(Mid[i] - min(C_i)) + min(C_i) = \frac{x-a}{b-a}(c-a) + a$$

[1] In the algorithms the list data structure is utilized instead of sets. I haven't been able to find any reliable source indicating the standard pseudocode representation of lists. I've encountered two examples where they were written with set representation and that's what I have done in this document as well.

input : $\mathcal{D}$ (instances set), k (# of clusters))
output: $\mathcal{D}$ (normalized instances set)

$cluster\_counter = 0$;
Clusters $\leftarrow \emptyset$;
foreach $d_i \in \mathcal{D}$ **do**
  $newCluster \leftarrow \{d_i\}$;
  $instance\_counter(newCluster) = 1$;
  $Clusters \leftarrow Clusters \cup newCluster$;
  $cluster\_counter++$;
end
foreach $C_i, C_j \in Clusters$ **do**
  $Compute \bigtriangleup(C_i, C_j)$
end
while $cluster\_counter \neq k$ **do**
  (C_m, C_n) = two clusters closest together in $Clusters$;
  $Clusters \leftarrow Clusters - \{C_m\} - \{C_n\} \cup \{C_m \cup C_n\}$;
  $cluster\_counter--$;
  foreach $C \in Clusters$ **do**
    $Compute \bigtriangleup(C, (C_m \cup C_n))$;
  end
end
Mid $\leftarrow \emptyset$;
foreach $C_i, C_{i+1} \in Clusters$ **do**
  $Mid \leftarrow Mid \cup \{(max(C_i) + min(C_{i+1})/2)\}$;
end
foreach $C_i \in Clusters \backslash C_k$ **do**
  foreach $d \in C_i$ **do**
    $d \leftarrow \frac{d - min(C_i)}{max(C_i) - min(C_i)}(Mid[i] - min(C_i)) + min(C_i)$;
  end
end

Algorithm 1: Single Linked Normalizing Algorithm

where x is the datapoint that goes under normalization, $min(C_i)$(=a) and $max(C_i)$(=b) are the minumum and maximum elements(respectively) of the cluster C_i that x belongs to, $Mid[i]$(=c) is the middle point amongst cluster C_i and C_{i+1}.

As mentioned, for the upcoming proofs, except Proposition 5 (regarding Property 6) we will work with this objective(update) function. Proposition 5 is perhaps the trickiest of all, in which we will be working on the clustering segment of the algorithm.

Lemma 1. *Single-linked normalization satisfies Property 1.*

Proof. Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a SAF, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ a semantic framework, $\mathcal{D} = \{M(a) | a \in \mathcal{A}\}$ be the multiset of model evaluations of all arguments, $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ the function that single-linked normalization algorithm implements and some arbitrary $k \in \mathbb{Z}^+$.

Let $x \in \mathcal{D}$ s.t. $x = 0$.

Since $x = 0$, it should follow that $x \in C_0$ (As zero is the smallest value in the unit interval).

$$\begin{aligned} \sigma_{\mathcal{D}}(0) &= \frac{-a}{b-a}(c-a) + a \\ &= \frac{0}{b}c && (min(C_0) = a = 0) \\ &= 0 \ \square \end{aligned}$$

Lemma 2. *Single-linked normalization satisfies Property 2.*

Proof. Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a SAF, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ a semantic framework, $\mathcal{D} = \{M(a) | a \in \mathcal{A}\}$ be the multiset of model evaluations of all arguments, $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ the function that single-linked normalization algorithm implements, some arbitrary $k \in \mathbb{Z}^+$ and assume $\nexists a \in \mathcal{A}$ s.t. $M(a) = 0$.

Property states that $x \neq 0 \Rightarrow f(x) \neq 0$.

We will use proof by contradiction. Let's assume $x \neq 0$ and $\Rightarrow \sigma_{\mathcal{D}}(x) = 0$. Then

$$\begin{aligned} \sigma_{\mathcal{D}}(x) = \frac{x-a}{b-a}(c-a) + a = 0 \\ \Rightarrow \frac{x-a}{b-a}(c-a) = -a \\ \Rightarrow x - a = -a\frac{b-a}{c-a} \\ \Rightarrow x = -a\frac{b-a}{c-a} + a \\ \Rightarrow x = -a(1 - \frac{b-a}{c-a}) \end{aligned}$$

We know that $1 \geq c \geq b \geq a \geq 0$ (i.e. $\frac{min(C_{i+1})+max(C_i)}{2} \geq max(C_i) \geq min(C_i)$). Thus $1 \geq \dfrac{b-a}{c-a} \geq 0$ (for $c \neq a$), and by that we can conclude that in the equality above the value in the parentheses is zero or positive. And finally from that we may conclude x is either zero or negative.

However x cannot be negative since $x \in [0, 1]$ as all datapoints. Furthermore x cannot be zero since our initial assumption was such that $x \neq 0$.

All in all the initial assumption lead to a **contradiction**.

□

Lemma 3. *Single-linked normalization satisfies Property 3.*

Proof. Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a SAF, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ a semantic framework, $\mathcal{D} = \{M(a) | a \in \mathcal{A}\}$ be the multiset of model evaluations of all arguments, $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ the function that single-linked normalization algorithm implements and some arbitrary $k \in \mathbb{Z}^+$.

To prove that the objective function is non-decreasing, we should simply show that the derivative is non-negative.

$$\begin{aligned} \frac{d}{d(x)}\sigma_{\mathcal{D}}(x) &= \frac{d}{d(x)}\frac{x-a}{b-a}(c-a)+a \\ &= \frac{b-a}{c-a} \geq 0 \qquad (1 \geq c \geq b \geq a \geq 0) \end{aligned}$$

□

Lemma 4. *Single-linked normalization satisfies Property 4.*

Proof. We may basically use the proof of Property 3. There we had proven that the objective function f is non-decreasing (i.e. $\dfrac{d}{d(x)}\sigma_{\mathcal{D}}(x) \geq 0$).

Thus under the same assumptions, it follows that $x_1 \geq x_2 \implies \sigma_{\mathcal{D}}(x_1) \geq \sigma_{\mathcal{D}}(x_2)$.

□

Lemma 5. *Single-linked normalization satisfies Property 6.*

Proof. In a nutshell in this proof we want to demonstrate that the clustering that Single-linked normalization computes has spacing at least as large as any arbitrary clustering possess.

Let $\mathcal{C} = C_1, ..., C_k$ be the clustering that Single-linked normalization outputs with spacing (i.e. distance between the closest pair of separated datapoint) S.

And let $\mathcal{C}' = C'_1, ..., C'_k$ some arbitrary clustering and has spacing S'.

So once again our goal is to show that $S \geq S'$. And we may accomplish this by displaying a pair of points separated by this clustering such that the distance between those separated points is S or smaller.

Case 1: The trivial case is when C_i' 's are the same as the C_i 's, possibly up to a renaming. Thus of course exactly the same pairs of points are separated into each of the clustering, so that the spacing is identical.

Case 2: The intriguing case is when the two clusterings fundamentally differ. Here we're going to argue that, for any clustering which is not merely a permutation of $\mathcal{C}$, there has to be a pair of points which is classified differently in the C_i' 's relative to the C_i 's.

By differently we mean that it should be possible to find a point pair p, q such that p, q belong to the same cluster C_i in $\mathcal{C}$. But in the alternative clustering $\mathcal{C}'$ they are placed in different clusters, say C_i', C_j'.

So here we may divide the proof into an easy case and a trickier case.

We know that Single-linked normalization works in a greedy sense, i.e. the separated pair of points that are closest to each other are the ones that should get merged. So for this reason, because it's always the closest separated pair that get merged, if we examine at the sequence of point pairs that get merged together, that determine the spacing in each subsequent iteration, the distances between these sort of worst separated points is only increasing over time. And this sequence culminates with the final spacing S of the algorithm. In other words, the spacing of the output of the greedy algorithm is the distance between the two clusters that would get merged if we ran the algorithm one more iteration. Thus we know that for any directly merged two points x, y, the distance $\bigtriangleup(x, y) \leq S$.

The easy case occurs the aforementioned point pair p, q is directly merged at some point. So our algorithm picked them at some iteration (since they constituted the shortest distance amongst separated points)., and their respective clusters were merged.

Then as we have just established, the distance amongst p and q should be no more than S. However since p and q are contained in different clusters in $\mathcal{C}'$, they are indeed *separated points* by definition, and thus they constitute an upper bound on S'.

Hence we have: $S' \leq \bigtriangleup(p, q) \leq S$. Therefor we've concluded the proof for the easier case of Case 2.

The tricky case occurs when p and q are indirectly merged, through a series of direct merges... □

Lemma 6. *Single-linked normalization satisfies Property 7.*

Proof. Let $F = \langle \mathcal{A}, \mathcal{R}, V \rangle$ be a SAF, $\mathcal{S} = \langle L, \curlywedge_{\mathcal{A}}, \curlywedge_{\mathcal{R}}, \curlyvee, \neg, \tau \rangle$ a semantic framework, $\mathcal{D} = \{M(a) | a \in \mathcal{A}\}$ be the multiset of model evaluations of all arguments, $\sigma_{\mathcal{D}} : \mathcal{D} \to [0, 1]$ the function that single-linked normalization algorithm(SLN) implements, $\bigtriangleup(x, y) \mapsto |x - y|$ be the metric SLN utilizes, $\mathcal{C}$ be the clustering SNL produces and some arbitrary $k \in \mathbb{Z}^+$.

Assume we take four arbitrary datapoints $x_1, x_2, x_3, x_4 \in C$ from an arbitrary $Cin\mathcal{C}$, for which the following statement holds:

$$
\begin{aligned}
&\bigwedge(x_1, x_2) \geq \bigwedge(x_3, x_4) \\
&\Rightarrow |x_1 - x_2| \geq |x_3 - x_4| \\
&\Rightarrow |x_1 - x_2 - (a - a)| \geq |x_3 - x_4 - (a - a)| && (a - a = 0) \\
&\Rightarrow |(x_1 - a) - (x_2 - a)| \geq |(x_3 - a) - (x_4 - a)| \\
&\Rightarrow |(x_1 - a)\frac{(c-a)}{(b-a)} - x_2 - a)\frac{(c-a)}{(b-a)}| \geq |(x_3 - a)\frac{(c-a)}{(b-a)} - (x_4 - a)\frac{(c-a)}{(b-a)}| \\
& && (\tfrac{(c-a)}{(b-a)} > 0) \\
&\Rightarrow |(\sigma_{\mathcal{D}}(x_1) - \sigma_{\mathcal{D}}(x_2))| \geq |(\sigma_{\mathcal{D}}(x_3), \sigma_{\mathcal{D}}(x_4))| && (\frac{x-a}{b-a}(c-a) = \sigma_{\mathcal{D}}(x)) \\
&\Rightarrow \bigwedge(\sigma_{\mathcal{D}}(x_1), \sigma_{\mathcal{D}}(x_2)) \geq \bigwedge(\sigma_{\mathcal{D}}(x_3), \sigma_{\mathcal{D}}(x_4))
\end{aligned}
$$

□

Theorem 1. *Single-linked normalization is a desired mapping.*

Proof.

- Firstly via Lemma 5 we have shown that Single-linked normalization has maximum-spacing. That was the only property we had to prove regarding to the labeling phase.
- By Remark 1, we had demonstrated the mapping implemented by the Single-linked normalization. In Lemma 1,2,3,4 and 6 we have proved that the particular mapping satisfies properties 1,2,3,4 and 7 respectively.

In conjunction, Single-linked normalization satisfied the list of properties considering the class of desired mappings. □

Proposition 1. *Single-linked normalization has quadratic time-complexity with respect to the number of datapoints (i.e. $\mathcal{O}(|\mathcal{D}|^2)$).*

Proof. We start the algorithm by creating a distinct cluster from each algorithm, and instantiating the initial cluster list. The time complexity for this procedure is linear wrt. datapoints, i.e. $\mathcal{O}(|\mathcal{D}|)$.

For each pair of points, the distance (metric) is computed. This is indeed an expensive procedure in the algorithm as it has quadratic ($\mathcal{O}(|\mathcal{D}|^2)$).

Then closest clusters are merged till the respective input constant value equals the cardinality of the cluster list. This procedure is linear in terms of cluster number. But furthermore at each iteration, the distances with the rest of clusters is computed for the newly merged cluster (again linear). Nested loops result into quadratic behavior ($\mathcal{O}(k) * \mathcal{O}(|\mathcal{D}|) = \mathcal{O}(k|\mathcal{D}|)$).

Finally we have two more separate linear procedures, first for computing the middle point amongst consecutive clusters. The second for normalizing each datapoints within its respective clusters.

All in all we have: $\mathcal{O}(|\mathcal{D}|)+\mathcal{O}(|\mathcal{D}|^2)+\mathcal{O}(k|\mathcal{D}|)+\mathcal{O}(|\mathcal{D}|)+\mathcal{O}(|\mathcal{D}|) = \mathcal{O}(k|\mathcal{D}|+|\mathcal{D}|^2) = \mathcal{O}(|\mathcal{D}|^2)$ (As $|\mathcal{D}| \geq k$).

□

2.3.2 Furthest normalization

The algorithm is pretty similar to the prior one, barring the merge condition in the labeling phase. Here in order to select the next cluster pair that's to be merged, the furthest datapoints in the clusters are considered rather than the closest ones.

In order to think about this in a different way, imagine the datapoints as nodes of a graph and assume that there are edges between the nodes if the distance between the datapoints is smaller than some threshold α. In graph theoretic terminology, in Single-linked normalization algorithm(the max-spacing) we were looking for connected components, and now in the new one we are looking for cliques.

Thus Furthest normalization algorithm is not max-spacing, but on the other hand it results into more compact clusters than before.

Conjecture 2. *Let* $d_1, d_2 \in C_i$. $\nexists d_3$ *s.t.* $\bigtriangleup(d_1, d_3) < \bigtriangleup(d_1, d_2)$ *where* $d_3 \in C_j$, $C_i \neq C_j$.

2.3.3 Additive normalization

... here our guiding principle is assigning datapoints one at a time to existing clusters, when possible...

Proposition 2. *Additive normalization has time-complexity of* $\mathcal{O}(k|\mathcal{D}|)$.

Proof. In the main loop of the algorithm we iterate over all of the datapoints ($\mathcal{O}(|\mathcal{D}|)$). For each point, in the worst case we have to check all existing clusters ($\mathcal{O}(k)$). Thus the time-complexity of the big for-loop is $\mathcal{O}(k|\mathcal{D}|)$.

As in Algorithm 1, we follow the same procedure for normalizing the values of datapoints within clusters. Hence again we have two isolated procedures that take linear time in terms of the cardinality of the dataset($\mathcal{O}(|\mathcal{D}|)$).

The total time-complexity: $\mathcal{O}(k|\mathcal{D}|) + \mathcal{O}(|\mathcal{D}|) + \mathcal{O}(|\mathcal{D}|) = \mathcal{O}(k|\mathcal{D}|)$.

□

2.3.4 K-means normalization

...flexible via termination condition, efficient and simple but not space-maximizing...

Proposition 3. *K-means normalization normalization has time-complexity of* $\mathcal{O}(t(k + |\mathcal{D}|))$.

Proof. The main while loop iterates till the termination condition is satisfied ($\mathcal{O}(t)$).

Inside the while loop we have two isolated procedures. First one iterates over each datapoint. Combined with the outer loop, we get $\mathcal{O}(t\mathcal{D})$.

The second iterates over all clusters and updates the mean info. Again combined with the outer loop, we have time complexity of $\mathcal{O}(tk)$.

input : $\mathcal{D}$ (instances set), k (# of clusters))
output: $\mathcal{D}$ (normalized instances set)

$cluster\_counter = 0$;
Clusters $\leftarrow \emptyset$;
foreach $d_i \in \mathcal{D}$ **do**
  $newCluster \leftarrow \{d_i\}$;
  $instance\_counter(newCluster) = 1$;
  $Clusters \leftarrow Clusters \cup newCluster$;
  $cluster\_counter++$;
end
foreach $C_i, C_j \in Clusters$ **do**
  $Compute \bigtriangleup(C_i, C_j)$
end
while $cluster\_counter \neq k$ **do**
  (C_m, C_n) = two clusters closest together in $Clusters$ wrt. furthest elements;
  $Clusters \leftarrow Clusters - \{C_m\} - \{C_n\} \cup \{C_m \cup C_n\}$;
  $cluster\_counter--$;
  foreach $C \in Clusters$ **do**
    $Compute \bigtriangleup(C, (C_m \cup C_n))$;
  end
end
Mid $\leftarrow \emptyset$;
foreach $C_i, C_{i+1} \in Clusters$ **do**
  $Mid \leftarrow Mid \cup \{(max(C_i) + min(C_{i+1})/2)\}$;
end
foreach $C_i \in Clusters \backslash C_k$ **do**
  foreach $d \in C_i$ **do**
    $d \leftarrow \frac{d - min(C_i)}{max(C_i) - min(C_i)}(Mid[i] - min(C_i)) + min(C_i)$;
  end
end

Algorithm 2: Furthest Normalizing Algorithm

input : $\mathcal{D}$ (instances set), k (# of clusters), γ (threshold constant))
output: $\mathcal{D}$ (normalized instances set)

Clusters $\leftarrow \emptyset$;
foreach $d_i \in \mathcal{D}$ **do**
  $assigned(d_i) = false$;
  foreach $C \in Clusters$ **do**
    if $|d_i - centroid(C)| < \gamma$ **then**
      $updateCentroid(C)$;
      $instance\_counter(C) + +$; $assigned(d_i) = true$;
      $break$;
    end
  end
  if $!assigned(d_i)$ **then**
    $instance\_counter(newCluster) = 1$;
    $centroid(newCluster = d_i)$;
    $Clusters \leftarrow Clusters \cup newCluster$;
  end
end
Mid $\leftarrow \emptyset$;
foreach $C_i, C_{i+1} \in Clusters$ **do**
  $Mid \leftarrow Mid \cup \{(max(C_i) + min(C_{i+1})/2)\}$;
end
foreach $C_i \in Clusters \backslash C_k$ **do**
  foreach $d \in C_i$ **do**
    $d \leftarrow \frac{d - min(C_i)}{max(C_i) - min(C_i)}(Mid[i] - min(C_i)) + min(C_i)$;
  end
end

Algorithm 3: Additive Normalizing Algorithm

input : $\mathcal{D}$ (instances set), k (# of clusters), t (iteration constant))
output: $\mathcal{D}$ (normalized instances set)

assign k cluster centers $\mu_1, \mu_2, ..., \mu_k$ to clusters $C_1, C_2, ...C_k$;
while $t \neq 0$ **do**
  foreach $d_i \in \mathcal{D}$ **do**
    assign d_i to the cluster C which has the closest center(mean);
  end
  $Clusters \leftarrow \bigcup_{i=1}^{k} C_i$;
  foreach $C_i \in Clusters$ **do**
    $\mu_i \leftarrow \frac{1}{|C_i|} \sum_{j=1}^{|C_i|} d_j$
  end
  t–;
end
Mid $\leftarrow \emptyset$;
foreach $C_i, C_{i+1} \in Clusters \backslash C_k$ **do**
  $Mid \leftarrow Mid \cup \{(max(C_i) + min(C_{i+1})/2)\}$;
end
foreach $C_i \in Clusters$ **do**
  foreach $d \in C_i$ **do**
    $d \leftarrow \frac{d - min(C_i)}{max(C_i) - min(C_i)}(Mid[i] - min(C_i)) + min(C_i)$;
  end
end

Algorithm 4: K-means Normalizing Algorithm

As in Algorithm 1, we follow the same procedure for normalizing the values of datapoints within clusters. Hence again we have two isolated procedures that take linear time in terms of the cardinality of the dataset($\mathcal{O}(|\mathcal{D}|)$).

The total time-complexity: $\mathcal{O}(t|\mathcal{D}|)+\mathcal{O}(tk)+\mathcal{O}(|\mathcal{D}|)+\mathcal{O}(|\mathcal{D}|) = \mathcal{O}(tk+t|\mathcal{D}|)$. □

2.4 Discussion

COMPARISON OF ALGORITHMS COMP/PROP AND GENERAL REMARKS

...the merit of the Single-linked normalization algorithm is easily displayed via property 6. for the other two, the properties that they adhere to constitute a rather flexible class of mappings. so perhaps the merit of these algorithms is not so easily seen. however they create a better valuation for the data. "better how", this should be captured by some property or parameter that can be measured.