Automated feedback generator for sketch-based open-ended questions

First Author FIRSTAUTHOR@LCS.MIT.EDU
Second Author SECONDAUTHOR@LCS.MIT.EDU

1. Abstract

Sketch-based open-ended questions deliver an accurate and adequate overview of students' understanding of a specific topic. They also ease the learning process, especially for children, by being a more interactive educational tool than the ordinary multiple-choice or written open-ended questions. This comes at a cost of a tedious and difficult process of correcting students' answers and providing personalized feedback. The challenges in providing automated feedback for such questions are summarised by the high computation process of sketch matching, the difficulties of extracting semantic meanings from unlabeled sketches, and the domain specificity of sketch-based questions. With all of these in mind, we designed a new pipeline for automatically providing verbal and visual personalized feedback of sketch-based questions. In order to use this method, only a reference sketch of a standard correct answer is required. The pipeline breaks down to five essential steps: combining strokes to form unlabeled, but fully meaningful objects, pair-wise registration between objects in the referenced sketch and the target sketch, finding the optimal one-to-one matching between these objects, exploiting the registration in object identification, and converting the visual object registration to a simple language.

2. Overview of the approach

In this paper, in order to provide personalized feedback, we follow an approach of registering the target sketch to a given correct reference sketch. This will help us recognize the differences between the student's answer and the reference answer. Depending on their semantic representations, these differences are later either accepted or eliminated by morphing the target sketch to match the reference sketch. This method will give us a visual automated feedback for free, which in turn is transformed into verbal feedback.

2.1 Object detection

TODO
Currently we are hard-writing which strokes belong to which objects. As we will be receiving a set of sketches to be evaluated, one possible approach is to run some well-known unsupervised object detector on these sketches.

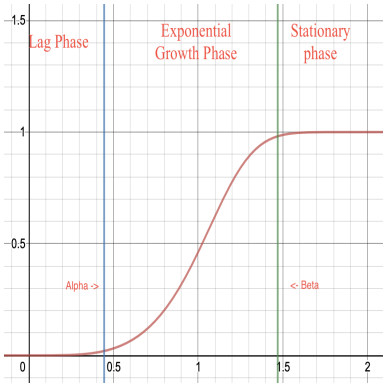

Figure 1. Mutated Sigmoid function

2.2 Object registration

We perform sketch registration by optimizing transformation parameters to minimize a dissimilarity function that measures the visual relation between the given objects, and the complexity of the transformation on the target sketch.

2.2.1 Dissimilarity function

For each point p in the referenced, and the target sketch, we add a penalty to the dissimilarity function as the euclidean distant d of the closest point to p from the opposite sketch (the target and the referenced sketch, respectively). Optimizing such a function would result in complicated transformations in favor of reaching the maximum possible registration. In order to have an acceptable registration with a minimum possible transformation, we overlook any distance d below a certain threshold A, and give a maximum (and almost equal) penalty to any distant larger than certain threshold B. Toward achieving that without affecting the smoothness of the dissimilarity function, we map the values of d that are less than A to the lag phase of the Sigmoid function, the values larger than B to the stationary phase, and the values inside the domain $[A, B]$ to the exponential growth phase. We used a mutated version of the Sigmoid function (1) with a domain $[0, \infty]$ and range $[0, 1]$, which is explained in Figure 1

$$S(x) = \frac{2e^{(x^n)}}{1 + e^{(x^n)}} - 1 \tag{1}$$

where n denotes the steepness of the increase of the penalty with respect to the increase in the distance d.

The value of d in the domain $[A, B]$ is mapped to the (α, β) coordinates, where α and β denotes the boundaries of the exponential growth phase of (2) and given by:

$$\alpha = \ln \frac{1 + C_1}{1 - C_1}^{\frac{1}{n}}, \beta = \ln \frac{1 + C_2}{1 - C_2}^{\frac{1}{n}} \tag{2}$$

where C_1 and, C_2 corresponds to the minimum and maximum penalty, respectfully.

In order to make the transformation understandable, we restrict the transformation to translation, shearing, rotation, and scaling. We also involved these transformations' parameters in the dissimilarity function that we are optimizing. Each of which is added as a penalty by multiplying their absolute values with some factor. These factors are used to make the module more robust to certain kind of transformations in favour to others. Thus, the transformation cost factors, along side with the steepness of the dissimilarity function n, are left to the user specification.
The final dissimilarity function of registering object O_1 over object O_2 becomes:

$$F(T)_{O_1,O_2} = \sum_{p \in O_1, O_2} \sigma S(\frac{(D(T(p)) - A) * (\beta - \alpha)}{B - A} + \alpha) + Cost(T) \tag{3}$$

Where T is the set of transformation parameters, T(p) is the tranformation of point p, D(p) is the euclidean distance of the nearest point to p from the opposite object, and σ is some factor.

2.2.2 Optimization process and complexity

Before we start the optimization process, we unify the number of points constructing each object, and we re-balance the distribution of these points by redrawing every object. We do that by walking equal distances over the strokes of the object.

The nearest point can be obtained using KD-tree. KD-trees are built at worst-case complexity of $O(nlogn)$ and querying the nearest point takes $O(logn)$ on average, which gives us a total complexity of $O(nlogn)$ per iteration. Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm is used for optimizing the dissimilarity function. This algorithm uses the first derivatives only and has proven good performance even for non-smooth optimizations. We combined this optimizer with the Basin-hopping technique for reaching the global minima. In practice, registering two objects of 70 points each takes on average 15 seconds on a normal Intel core i5 machine.

2.3 Optimal object matching

One assumption that we carry on through the process of matching is that the dissimilarity function (3) will give higher dissimilarity for any two semantically different objects than any two semantically similar objects. This is true as long as σ in (3) is set to a relatively large value (i.e better matching is favoured despite the more complicated transformation). The final dissimilarity between a pair of objects is gives by:

$$F_{ij} = \min_{p \in Pj} F(O_i, p) \tag{4}$$

Given the set of dissimilarity F_{ij} between all pairs of objects in the target, and the referenced sketches, we want to minimize the total dissimilarity of matching subject to the constraint that the matching is one-to-one

$$H(\pi) = \sum_i F(O_{1i}, O_{2\pi(i)}) \tag{5}$$

Where $\pi(i)$ is the object from the reference sketch corresponding to object i from the target sketch. As the matching is one-to-one, π should be a permutation. This problem is modeled as the weighted bipartite matching (or optimal assignment problem), which can be solved in $O(M^3)$ using the Hungarian algorithm, where m is the maximum number of objects of the target, or the referenced sketch. In case the number of objects is not balanced between the referenced and target sketch, possible dummy objects can be added with a dissimilarity of infinity to each of the other objects. The objects that are matched to the dummy objects are then vanish (if they belong to the target sketch), or appear from nonexistence (if they belong to the reference sketch)

The model of matching is designed such that it can be easily run under multiple processors, which reduced the running time dramatically. For a sketch of 5 objects, each of them has average 70 points, running on an Intel core i5 machine with 8 cores takes approx. 45 second to obtain the dissimilarity between every pair of objects and obtain as well the optimal matching.

2.4 Object identification

After performing the optimal matching between the objects, the context similarity of the matched objects needs to be confirmed or rejected. We identify the similarity of the matched objects by performing prototype-based recognition. We measure the distance $D(o, p)$, which we will make precise shortly, between the object and its corresponding prototype. $\min_{p \in P} D(o, p)$ gives us a scale for confirming the similarity of the objects and apply the corresponding transformations, or declining them and morph that target object to the nearest prototype.

2.4.1 OBJECT CONTEXT DISTANCE

On the assumption that we restructured the objects to the same number of points distributed balancedly, object registration allows us to extract correspondences between the points of the pair of objects. This in turn allows us to correspond the strokes of the pair of objects to each other. Multiple strokes might need to be combined in order to corresponds to a single stroke in the opposite object. Following a similar approach to what we have presented in 2.2, we perform registration between corresponding strokes but with eliminating the cost of transformation $Cost(T)$ from the dissimilarity function (3). This will result in a maximum possible matching between the registered strokes. We define the context distance between any two objects O_1, and O_2 as the total dissimilarity between the registered strokes plus the total differences of the turning angles between any two consecutive strokes.

$$D(O_1, O_2) = \sum_{S_i, S_j \in O_1, O_2} F(S_i, S_j) + abs(A(S_i) - A(S_j)) \tag{6}$$

Where $A(S_k)$ is the turning angle between the strokes S_k and S_{k-1}.

2.5 Extracting verbal feedback

TODO

3. Future Work

- I have not yet implemented multiple prototypes handling.
- To increase the accuracy of the identification phase we can work on feature-based instead of stroke-based. Meaning that instead of corresponding strokes to each other, we can extract features from the objects and corresponding these features to each other. We can extract these features based on the time elapsed during drawing. The idea is that people tend to wait a bit while moving from one feature to another. We can track this time elapse to extract features and perform registration upon these features.
- Another idea for increasing the accuracy of the identification phase would be detecting object outliers (objects which we are not very sure whether they match the corresponding object or not). Then we can manually accept or reject such objects and add them (in case of confirmation) to the prototypes set of the corresponding object.
- Suppose we have three circles in the target sketch, and three other circles in the reference sketch, these circles might match to each other in a different order. That will result in unnecessary feedback. We need to find a way to include the spatial relation between the objects in the matching algorithm.
- Reduce the running time