Clustered Text: A Never-Ending Story

Walter B. Gress V

January 6, 2020

Clustered Text uses a combination of K-Means,

Decision Tree, and hard-coded data to perform

unsupervised learning over the domain of text.

# introduction

Clustered Text performs a mining operation to get a domain to work with..

# in detail

The algorithm starts by pulling in data from one hundred newsgroups. It then pulls in data from H.G. Wells and finally text Isaac Asimov to complete the domain. Once it has the unlabeled data to work with, the features are marked 1 for the first, 2 for the second, etc…until you reach 1,000 or more features. Because of the sheer size of the data set, we are using semi-supervised learning. A set, say 50 features, are labeled by hand with annotation representing connections or a “storyline”. That set is used to train the remaining. K-Means is used to cluster data.

The k-means data are used in an hierarchy manner. Once a set of clusters is formulated, they are used to form a tree. The y-axis is the individual text sets and the x-axis is an enum. Examples of enums might be GENRE={ Horror, Mystery, Science Fiction, then the next level of the tree might be the enum SCIFI = { Laser, Spaceship, Moon, Spacestation } and choose Spacestation and we can drill down further to another node in the tree = “Scientist” and next might be the EMOTION={ Happy, Angry, Tired, In Love }Let’s choose Tired.

Now we know that the agent is in a science fiction world, there is a spaceship, there is a scientist and he’s tired. As the algorithm drills down it saves the information from the enum and uses it to construct the story.

# More Training

So we have some data about the agent. More to include a flow of storyline we have to have a fresh storyline. We need to recycle the annotated text before run k-means. So we run TRAMS. We choose a paragraph of text from the labeled corpus and insert variables from initial into annotated locations. This continues

until game is terminated.

# in brief

The algorithm pulls an annotated text corpus of text from a wide range of dialogue. The annotation keeps the link from paragraph to paragraph is maintained. A single paragraph is taken from the corpus. Preprocessing occurs

where the annotated part of the text matches up. K-Means is used to cluster, y-axis samples from a paragraph one or several text and enums through – one set of enums. And choosing one enum opens up more enums. The information is captured and stored.

# transition paragraphs

1. Pull nouns, characters
2. Fill annotations (characters, nouns)
3. Maintain these character information
4. Take the last half of the last paragraph and the next part of the next paragraph and “blend” them.

# transition in detail

After we have corpus in detail and partially annotated, etc. we have to transition. This is the novel part of the algorithm.

1. First step is to pull the annotated data. Check for the annotated data on the first paragraph and the next (new) paragraph. Fill in the annotated data with the preprepared data.
2. The last several sentences are grouped with the first several sentences of the next paragraph. This is treated as a paragraph on its own.
3. Perform TRAMS on the group paragraph. Perform k-means, using unsupervised learning using the last iteration of the algorithm, that is, the last paragraph, for training.
4. Perform TRAM on the next paragraph. Perform k-means, using unsupervised learning using the previous iterations of the algorithm as a training set.

# clustered data in detail

The y-axis has numbers that represent represented text. Each text could be like, for example, “Fireball!” or “I’m afraid we are closed for the night.” Two make this make more sense, we have characters. Each character has a role. For example, a Mage, or a Knight. These enums are structured in a decision tree. There is also an enum of Era so we know what time period things are happening. The enums are on the x axis and the labels are on the y-axis.

Because we don’t want everything to seem scripted, enums are chosen at random. The root

node is randomized as well (The era enum).

# incorporating enum data into corpus text

1. So, we have all this domain specific knowledge to the storyline. What do we do with it?
2. Do we need to place the domain specific knowledge (Fitting) into the annotations?
3. For the unlabeled data how do you substitute the domain knowledge?
4. Split the train data again and determine where the domain data fits.
5. How do we know where the data fits?
6. Maybe we don’t. The enums and the decision “direct” the storyline. The only data we get out of the decision tree is the text and the text (the model) is already trained.

# summary

1. Randomize data for annotation
   1. Data should be related
2. Pull the very first paragraph
3. Perform TRAMS on first paragraph
4. There is already m=50 labels annotated
5. Perform k-means semisupervised clustering using the m labels that are already labeled.
6. Randomize x-axis. Generate the features on the x-axis and then randomize a choice on the x-axis. Then follow the y-axis until you reach a cluster.
7. Follow the decision tree with the enums. Store data as you follow.
8. Perform the fitting process.
9. Output a paragraph of generated text
10. Prepare for the second paragraph. Fill the annotated data.
11. Form a transition paragraph out of the last paragraph and upcoming paragraph. Make this the group paragraph.
12. Perform TRAMS on the group paragraph
13. Perform k-means on the group paragraph, using the labels from the last paragraph. Output group paragraph.
14. Perform TRAMS on the upcoming paragraph.
15. Perform K-means on the upcoming paragraph.
16. Iterate through decision tree to produce the next output and repeat this process.

# kmeans clustering and clustered text

1. KMeans is an unsupervised classification algorithm. The data points

are measured as such. The y-axis (the labels) or output is a phrase or piece of text. The independent variable, x, spans over an enum’s value. We know this data point randomly chosen represents an enum value and a text string. It also needs to have a clustering value.

1. First we need to know our k value.
2. Guess centroids for all the clusters.
3. Calculate, for each data point, the Euclidean distance to each centroid, and assigns the point to the closest cluster.
4. Centroids are calculated again.
5. Points are calculated again and distances are as well.
6. Process repeats until centroid stop shifting around and converge.

Once the clusters are grouped together they are assigned to the root node of a decision tree. The next step is to go down the tree by a level and plot the random enums and load a new paragraph along with a new group paragraph.

The key is to have a) a large of corpus of narrative b) a large number of useful TRAMS to revolve the paragraphs.