

Data Driven Approaches: A Basis of Everything

- Rohit Kumar -
(rohit@iiit.net)

Can't say if this syndrome hits everyone, but since my recent experience with research, I have had a very one track mind. Everything I think seems to lead me to the same approach or possibly I am being unable to think beyond it. But since I've been thinking a lot about Data Driven approaches and Unit Selection, I felt that they could be a solution to a large number of problems and may well depict the way humans work. So here are some extrapolations and imaginations that strike me.

We can classify all mundane tasks and a large number of other tasks and problems fundamentally into two categories.

1. Recognition

2. Synthesis

Here we can take recognition synonymous to reading, matching and even understanding and lots of other such words. Similarly we can take synthesis synonymous with generation, production and other words.

Commonest examples in the recognition class of problems being speech recognition, hand writing recognition, pattern recognitions as in face recognition, speaker verification, language identification, language understanding, stock prediction, etc. None of the problems have well defined solutions and all are performed very well by humans as compared to machines.

In the synthesis domain, we have speech synthesis, facial animation, language generation, music synthesis, hand writing generation. Though I have mentioned a very small set here due to limitations of depth and breadth of my experiences, but I have a instinct that tells me we can classify most of the computing problems (studied in relation with AI) into these 2 domains.

Data Driven approaches used to solve some of problems mentioned have proven to be highly successful and quite economic in present scenario considering exponential fall in the costs of computing resources. Probably interested people should read Dr. R. Reddy's Newell Simon Lecture to further see how deep the implications of the dream of infinite memory and infinite speed are.

Data Driven approaches to solve these problems is based on use of large amount of data that acts as a reference set from which the system derives information and knowledge in some suitable manner to solve the problem at hand.

Now what are these suitable methods? This is exactly where the 2 classes of problems split. The suitable method for each of these problems is different. But still within a single domain the method has a global application with only changes very specific to the problem. Moreover the 2 different approaches themselves are related. (as I will talk about later on in this article)

Basic synthesis approach is unit selection of appropriate units from a manually and/or automatically annotated set of data. Stepwise synthesis algorithm is as follows.

Database Creation:

- Collection of Data: Involves gathering and pruning of appropriate data while minimizing noise elements and maximizing useful variety of contents.
- Annotation of Data: Involves Segmentation and feature extraction
- Storing in appropriate form: Selecting a proper representation

Actual Synthesis

- Description of desired synthesis: Using a representation comparable to the representation selected above
- Generating a list of possible candidates units from the database for each unit in the desired synthesis
- Selecting the most appropriate units by optimizing a distance measure between the desired unit and available units
- Putting the selected units together so that they fit together without discontinuities

There are issues which are problem specific that need to be handled to use the above approach. The basic issues are

1. What should you have in your database? The generic reply is units of whatever you are synthesizing.
2. What features should we look for? Look for features that are characteristic and distinguishing.
3. How do we extract the desired features? Depends on the feature
4. What is the appropriate representation? Depends on the type and number of units and features. Generically the representation should allow categorization and speedy search.
5. How do we generate the description of desired output? Depends on what you are synthesizing and what representation you are using for storing in your database.
6. What kind of distance measure do we use? Depends on the problem. But it would be some kind of weighted accumulation of Euclidean distances in various feature dimensions.
7. Finally how do we put together the selected units? Depends on the kind of units we are working with

Now to depict the generality of the approach, I discuss the above issues specifically for 2 problems: Speech Synthesis and Facial Expression Generation.

Speech Synthesis:

1. Units: In speech you can have units at various levels of language like phones, diphones, syllables, words, phrases. But the kind of units you select should be elementary, easy to combine together smoothly i.e. very less co-articulation beyond the unit. Also the issue is what all units we should have. First we should have units enough to synthesis everything. Second we should have as much variety of units in various contexts as optimally possible.
2. Features: Speech has so many features. So you have a lot of choice. But the features that you use should characterize the units. For example the vowel in a syllable unit is a very significant feature. Similarly duration of the unit, frequency contours are all significant features. There choice is an arguable issue.
3. Feature Extraction: Quite established for speech features. But we need to manually segment the speech; though automatic segmentation is available, manual pruning is still required.
4. Appropriate Representation: Cluster the units on the basis of a distance measure. Refer to Dr. Alan Black's Festival System and its Unit Clustering algorithm.
5. Description of desired output: Sequence of desired units with provision for finding the contextual features and desired features.
6. Distance measure: The distance measure suggested by Mr. S. P. Kishore is very general and can be used in lot of applications. We need to come up with optimal weights and what all features do we need to match.
7. Concatenation: Depends on what units you are using. Simple Zero Crossing trimming can do well in case of larger units. From smaller units like phones and diphones, edge smoothing, both in amplitude as well as frequency domain could do the trick.

Facial Expression Generation:

1. Units: Chunks of face parts. We should keep a basic frame of face and store a variety of the features that change while speaking like lips, cheeks, forehead, eyeballs, eyebrows and chin. These units should be available in variety of situations. The frame of face should be such that combining these pieces on the face should be simple.

2. Features: There are several features. The word being spoken, phone being uttered, mood of expression, etc. But the trick is in choosing the best of these. By simple analysis you can conclude that the shape of the face is to a major extent guided by the vowel being spoken. Try it. Also to some extent it is dependant upon the place of articulation of the phone. Occasionally there are flicks like eyebrow movement, eyelid flicking, crowding of lines on forehead, a stop after some words, sigh after a long sentence and finally an umm, ah, in between randomly. Well OK not very randomly but at the end of phrases. They these are cosmetics. Basic feature is how the lips cheek and chin changes.
3. Feature Extraction: I don't know how the Project Oxygen team at MIT does it, but I feel it has to be done manually atleast for now. Availability of proper software tools can ease the manual work to quite an extent.
4. Appropriate Representation: The units may as above be clustered up for easy and speedy retrieval given the desired features.
5. Description of desired output: Sequence of desired facial expression with features like shape of eyes, lips, cheeks etc. This sequence should be automatically generated from a given text sentence for which the facial expression needs to be generated. The cosmetic elements may be added into the specification after the basic elements of the specification are available. The description of desired output should also contain the duration of time for which a particular expression should stay. This information can be derived from study of spoken corpus again by a data-driven approach.
6. Distance measure: The distance measure as referred to above, with parameters like duration can be used.
7. Putting together the pieces: Now this is a bit of an issue in this particular problem though morphing and things are available, exact placement of the component on the dummy face will be an issue. One solution can be including the data regarding the positioning of a component on the dummy face being included with the features. Also transition from one facial expression to another without a jerky effect will be an issue.

Further I see that it is fairly simple to derive solutions similar to these for other synthesis domain problems like hand writing synthesis and even language generation. Another issue is how much Data should the data driven approach use. This is very problem specific issue of analysis. But there is always an optimal size. Beyond that the data is contributing no more than junk. For example, Dr. Victor Zue notes that a recording of 20 – 30 minutes is sufficient to generate automated facial expressions. We have used a corpus of around 45 – 50 minutes in speech synthesis. But observations say that, even a corpus of 25 – 30 minutes would have done equally good. To estimate optimal size of a corpus is vastly explorable domain.

So with this I end the discussion of generalized synthesis approach and proceed into recognition of which I lack experience but have ideas.

Let me define the generalized recognition problem. All recognition problem reduce to matching a given feature vector to a very long sequence of features in the data. Now first issue is that the features in most of the recognition domain problems will not match accurately into the data. So we need a measure of distance between a feature vector and its probable match. Second problem in matching the feature vector would be that not all elements of the vector will be significant. Some will be replicates of adjoining elements and other would be noise elements.

Leave aside these mathematical issues of how to match 2 feature vectors (I am sure there are existing solutions to that which I haven't happened to have studied yet: Markov Models, Regression or Mahalanobis Distances etc.) we are faced with other problems. As long the number of elements in the feature vector is limited and small, brute force match with some one to one distance measure can be satisfactory. But as the number of elements in the feature vector grows the problem becomes computationally explosive.

As we did in the case of Synthesis domain problems let us try to formalize the approach for recognition domain problems. As in the case of synthesis domain problems, a data driven solution to

generalized recognition problem involves building up a database. The same issues and the same steps are faced here. What units are we going to have in the database? What all features do we need to store? How do we get those features? How do we store the units? The generic answer to the last of these questions is that we store the units so as to facilitate the way we are going to use the units? So, how are we going to use the units?

In the recognition domain, we can use the databases in the first stage to build up models, rules, knowledge and go down to the approaches that use these elements for recognitions. In this case data driven approach is being used to generate the offline information needed for the online processing required during the recognition. The advantage of following this approach is that we can make use of well established knowledge based approaches for recognition. The use of data is eliminating the cumbersome part of knowledge based approach i.e. building up the knowledge itself. In this the data is not being used at runtime and is arguable to call the approach a data driven one to some extent. Another question mark in this way of using the data is Will the model or the rules or the knowledge we generate out of the data be able to represent all the diversity and precision available within the actual data? Possibly to a big extent; But will that be sufficient is another question, solution to which is very much dependant on the problem itself.

Another way of using the data for recognition could be using the whole database online. Given the input pattern (some feature vector) we search within the data to find a nearest match of the pattern. The problems involved with this approach is, Do we have a fast search algorithm for searching the pattern we are looking for? Fast is a relative term. It depends on the kind of machines we expect to use the algorithms. As the performance of computing machines keep increasing, the issue of speed is more or less subdued. So the next issue becomes as to how do we match i.e. what kind of a distance do we use to match a feature vector to another. There surely are mathematical solutions to this question, each having its own advantages and disadvantages.

So given an input pattern, the process of recognition involves:

1. Automatic segmentation of the pattern to elementary units
2. Generate the feature vectors of the sequence of segmented units
3. Recognition of the units by matching the feature vector to the those in the data
4. Interpretation of the results of recognition

The questions involved in using this approach to a particular solution are

1. Do we have an automatic method to segment the pattern into elementary units? If we do how accurate is it? Answers depend on the kind of pattern we are trying to segment. In most cases segmentation techniques are available but accuracy is still a problem
2. What are features should the features vectors contain? Again the answer is dependant on the kinds of units and the problem itself. But as in the case of synthesis problem, the features should be distinctive and characterizing
3. Do we have a proper algorithm for rapidly searching the data and finding the best match to the feature vector? How do we define the best match itself? What kind of a distance do we use?
4. How shall we interpret the results?

I would love to learn and write more about the generic answers to these questions and I will as my knowledge and understanding of the underlying issues of the various problems in the recognition domain expands. But for now, this is the end of the discussion on recognition domain problems. Further a couple of ideas and issues have been are put down.

One critical issue with all data driven approach based solutions that I would like to explicitly mention is involved in designing the content of the data. The choice of content that is going to be incorporated into the database to drive the system is motivated by several factors. There is always a need to optimize on size vs. performance while building up the database. Deciding on what the optimal size of the

database would be becomes an issue. Also the content of data should be well chosen to provide a wide coverage of the units in a variety of contexts.

As I say that data driven approaches could go on to be the solution to every problem, I visualize a machine that would be able to do all the jobs in the domains of recognition and synthesis. The design of such a machine will based on 2 generic algorithms one each for recognition and synthesis. For each problem at hand, the machine will only required to be configured with some parameters like

1. Specifying which domain the problem falls into
2. How does the machine get the relevant data to drive its generic algorithms?
3. What all features does the machine extract from the data and how does it extract the features? Feature extraction would be like plugins. The specification will involve names of plugins like Pitch, Mel Cepstral Coefficients, etc.
4. What kind of parameters does the machine use in the various generic distance measures?
5. What kind of search algorithms does the machine use?

I am sure such generic machines would come up in future. Generic frameworks will be built that will allow easy solution to any problem using standard approaches used to solve other problems. As such standard approaches are discovered our understanding of the basic processes of human mind will be broadened.

In my personal communication, I have received the following comments from Dr. R.I. Damper about my understanding and extrapolation of the use of Data Driven Approaches. Dr. Damper is a renowned researcher in the field of speech technology and editor of the paper collection "Data – Driven Techniques in Speech Synthesis," Kluwer Academic Publishers, 2001.

Comments from:

Dr. Robert I. Damper, Head, Image Speech and Intelligent Systems Group,
Department of Electronics and Computer Sc., University of Southampton, UK

The idea that human tasks are either "recognition" or "synthesis" is really only saying that when interacting with the environment, we take information in and produce actions or outputs. Then saying that both input and output are "data driven" is really taking sides in the age old debate about nature versus nurture, or what is innate versus what is learned. Basically, you are minimising the importance of what is innate and maximising the importance of what can be learned from data - or the environment. Philosophically, it's structuralism versus empiricism. The arguments for and against have raged back and forth for a long time.

References:

1. Dr. Raj Reddy, Newell Simon Lecture
2. A. J. Hunt, A. W. Black, "Unit Selection in a Concatenative Speech Synthesis System for a Large Speech Database," ICASSP, 1999
3. S. P. Kishore, Rohit Kumar, Rajeev Sangal, "A Data Driven Synthesis Approach for Indian Languages using Syllable as Basic Unit," ICON 2002
4. Tony Ezzat, Tomaso Poggio, "Videorealistic Talking Faces: A Morphing Approach," AVSPW, 1997