NFL Route Identification: Clustering Analysis

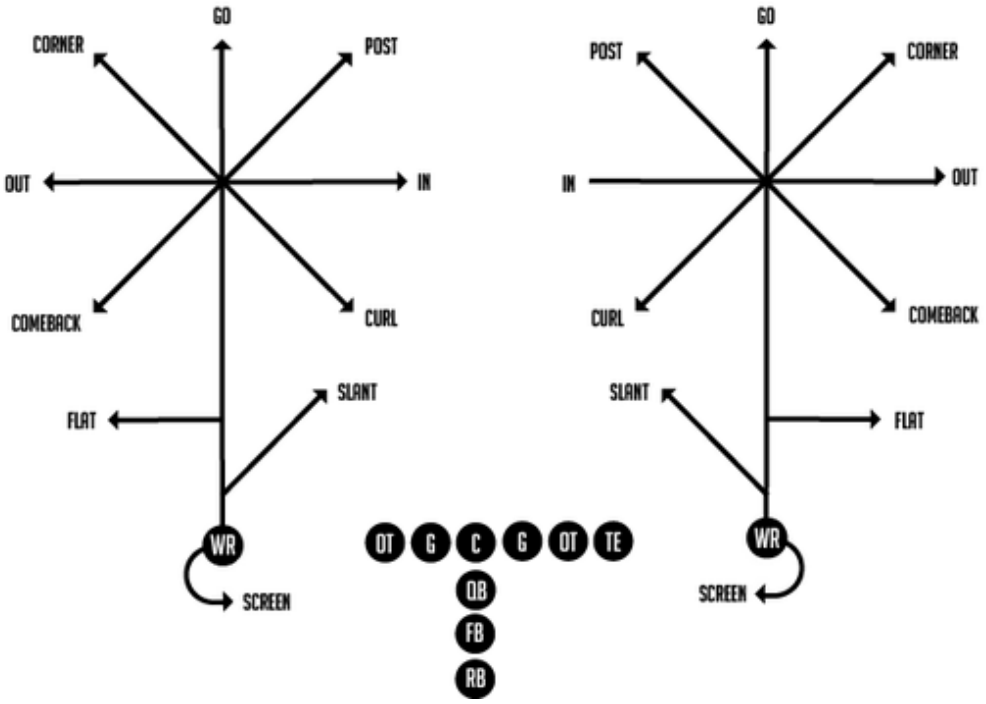
Dan Grogan - AMAT 585

The game of football has been long thought to contain too many variables to perform a thorough analysis on compared to other sports. While sports like baseball, which was thought to pioneer analytics, have had a head start in their analytical revolution, football's has seemingly just begun in recent years. My research project this semester is my attempt to join in the fun, as well as test a new dissimilarity metric that I have never heard of before. The goal I aimed to achieve is to accurately group NFL route runs into clusters. Before we get into what did and didn't work, we will go into the football background necessary in order to comprehend what I am attempting in this project.

Background

There are three phases in the game of American football: offense, defense, and special teams. Within the realm of offense, there are two types of plays, one of which is deemed as a passing play. In order to be classified as a passing play, a player (most often, however not always, the quarterback) must make a forward pass to an eligible receiver. Wide receiver is the position that is devoted to receiving forward passes, however running backs, fullbacks, tight ends, and even sometimes lineman (although this is much rarer) can receive the football as well. A typical passing play consists of multiple players running a route (a path that is predetermined by the design of whichever play has been called) with the intention of finding an opening on the defense in order to safely receive the ball from the quarterback and progress down the field.

Combined, there are considered to be a total of nine routes that make up what is known as the route tree. Each of these routes is given a name and a number. For instance, if a receiver runs in a straight line perpendicular to the line of scrimmage (the starting yard line in football), this is known as either a 9-route, a fly route, or a go route (these are all synonymous). On top of this, there are many variations of routes that an eligible receiver can run. For example, if you combine a slant route with a go route, you will obtain the aptly named "Sluggo" route. Below is a diagram of what the route tree looks like. We will not be considering such routes in our analysis.



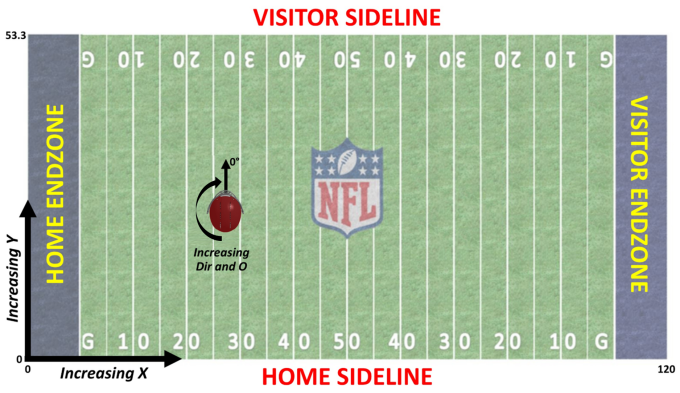
Having an algorithm to be able to cluster or classify NFL route runs can prove to be useful to NFL franchises. In order to best prepare for an upcoming game, coaches and players will study film of either their upcoming opponent or their own film from previous games. Defensive coaches or players may want to observe the opponent's tendencies when running all sorts of plays before their upcoming game. They could also want to correct defensive lapses against certain plays to better their own players. Offensive players and coaches might want to make improvements to the playbook or execution of plays by studying their own film. That said, the ability to query plays by which routes were run by your team or the upcoming opponent would provide to be useful to NFL clubs. Typically, plays and the routes run in them are manually labelled by team staff, which can be time-consuming. Having a way to automatically classify or cluster NFL route runs would benefit NFL clubs.

This task has already been successfully achieved by Chu et al. (2019). Using Bernstein Bases and Bézier Curves to essentially fit a multivariate linear regression matrix, and then creating a mixture model from the output, Chu was able to, for the most part, cluster NFL routes successfully. There were some problems with the methodology that I will get more into later. I will be working with the same data that was used for this method.

In 2013, the NFL partnered with Zebra Technologies in order to begin embedding Radio-Frequency-Identification (RFID) tags in the pads of NFL players. Every tenth of a second, these RFID tags capture information about a player's movement. This includes things like position on the field, distance travelled from prior time point, speed, acceleration, and more. Later, Wilson successfully tagged footballs with RFID tags and in 2017 the NFL partnered with Amazon Web Services in order to produce advanced analytics for the sport. The NFL produces copious amounts of player data from every game played that they have made available over the past three years in their annual NFL Big Data Bowl competition.

The dataset that I am choosing to work with, similar to the one Chu (2019) worked with, is from the second annual NFL Big Data Bowl. This data was made available on Kaggle.com in the form of 20 different datasets, each containing different forms of information. I will focus on the datasets containing plays from weeks 1 through 17. As the 2020 NFL Big Data Bowl was about passing plays, these datasets will contain every single time point captured for each offensive player in any given play that happened that week. Each time point has a label corresponding to which player the RFID tag was attached to, as well as information to distinguish which play each specific capture was related to. There are also key events labelled in each capture to indicate things such as the play's beginning, as well as some other key things that can happen during a play. Below is a snippet of what this data looks like.

time	2018-09-07T01:07:14.599Z
x	91.73
y	26.67
s	0.0
a	0.01
dis	0.02
o	289.57
dir	240.93
event	None
nflId	310.0
displayName	Matt Ryan
jerseyNumber	2.0
position	QB
frameId	1
team	away
gameId	2018090600
playId	75
playDirection	left
route	NaN



One thing to note is that I won't be considering data for players that are not running routes on a given play. Also, I will only be considering the following features in my analysis. Below is a list of which features I will use and a brief description of the feature:

- time: Timestamp of when data was captured
- x: x coordinate of the player's location at captured, ranging from 0 - 120 yards
- y: y coordinate of the player's location at captured, ranging from 0 - 53.5 yards
- event: a label for special events that happened at capture, such as ball snap
- nflId: a unique numeric ID for player tagged
- displayName: Name of the tagged player
- position: Position (or role) of the player (ie, quarterback, or 'QB')
- frameId: This number indicates that this is the n^{th} capture of the play
- gameId: unique numeric ID for the game
- playId: numeric ID for play, unique within the scope of each game
- playDirection: Direction in which the offense is moving
- Route: label of the route run by an offensive player (if they are running one)

Due to the nature of how this data is collected and stored, I have performed a number of different preprocessing steps to ensure accuracy in comparisons of routes. These will be very similar to the steps taken in Chu (2019). These steps are listed in detail below:

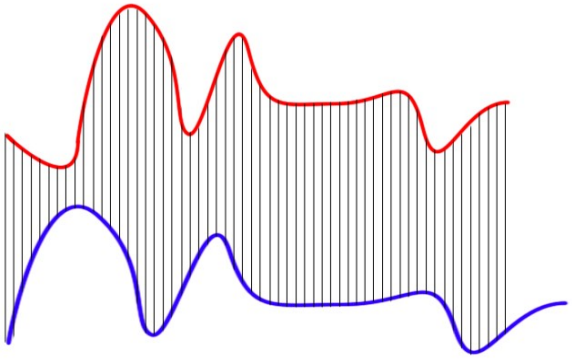
1) I will only consider a route to be the positional data captured between (and including) the events "ball snap" and any event indicating the end of a play (there are multiple events that can indicate this).
2) The time at which each data point is collected should be standardized. This is less important but I am of the opinion that it makes things neater or easier to understand.
3) Routes must be standardized. Routes start at different coordinate locations for every play. Routes will go in different directions depending on whether the offense is moving left or right down the field as well. Lastly, a route is flipped depending on which side of the ball snap it's on. For instance, an In route is a route that cuts toward the inside of the field. As routes can be run from both sides of where the ball is snapped, an In route runner starting from the left side of the field would make their cut (or sharp turn) to the right, while an In route starting on the right side of the field would make its cut to the left. We still want to label these the same route, so we must account for all of the aforementioned factors before we can perform our data analysis.

Once this preprocessing has been done, I can finally begin my data analysis. From what I am aware of, this is where I differentiate myself from methods that have previously been performed. What makes my analysis possible is the use of Dynamic Time Warping as a metric of dissimilarity between route runs. Once I have calculated the pairwise distance using Dynamic Time Warping, I will be able to perform three different analysis methods on my data. The first that I will perform is spectral clustering. Second, I will apply ToMATo clustering to my data to compare the differences. Lastly, I will look to embed my data using UMAP to see if I can find any natural clustering happening based on the labels that were given to each route from the original data.

Before I go into my methodology any further, I will discuss the essence of what Dynamic Time Warping is. In my analysis, I will be treating a route run as two different sets of time series data, one set for each coordinate used to represent player location. One of the problems with this data compared to other time series data is that the length in which a route is captured is not uniform, this is due to the fact that we are capturing our data at a constant rate but plays can last for varying amounts of time. Comparing time data of varying lengths can be tricky. Chu et al.

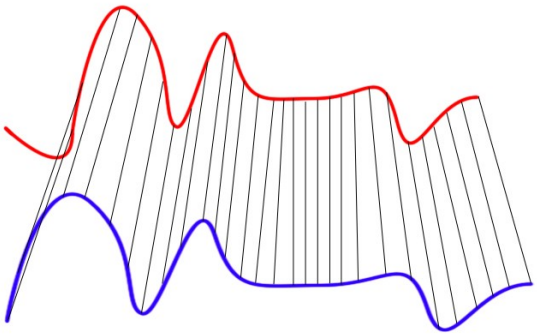
(2019) solved this issue by using Bézier curves, as those allowed for transforming the data into something that didn't require the varying time to be a factor. In my project, the way I correct this issue is using Dynamic Time Warping.

Dynamic Time Warping is a method of calculating dissimilarity between arrays of varying lengths. Without getting into formal definitions, the essence of what Dynamic Time Warping does is to learn an optimal mapping between two sequences that minimizes the absolute differences between each mapped indices and their values. There are a few constraints that must be followed. Every index in each sequence must be mapped to one or more index in the other sequence. The first index from both sequences must be mapped to each other; same with the last index in each sequence. Finally, the mapping of indices from one sequence to the other must be monotonically increasing. Here is a visual comparison of a Euclidean matching and a Dynamic Time Warping matching.



Euclidean Matching

The



Dynamic Time Warping Matching

final constraint may be the most important constraint for this analysis as it solves a crucial problem that was encountered in previous research. What Dani Chu encountered in their analysis was that their clustering algorithm had a difficult time differentiating between a go route and a comeback route. Functionally, their algorithm deems them to be nearly the same route. Dynamic Time Warping solves this issue due to the monotonicity of the process. The way indices are increasingly mapped forces the locational data to be compared similar between similar time steps. Thus, theoretically, when the player running a comeback route makes their cut and starts to move in the opposite direction of a go route, the dissimilarity between the two routes will start to grow larger.

These distance calculations will be made using the fastdtw package that is available in Python. For each pair of routes, I will perform Dynamic Time Warping on the arrays containing the x coordinates as well as on the arrays containing y coordinates. I will sum the distances obtained from these two calculations, creating what I consider the dissimilarity between the two routes. From there, a dissimilarity matrix can be constructed, which is the necessary input for the research methods that I mentioned earlier.

With this project, I aim to achieve at least one of the following goals. The first goal I wish to achieve is to obtain somewhat accurate clusters of routes. If routes get grouped similarly, I will be okay with that, as that gives me something to potentially work on moving forward if I plan to continue this analysis in the future. The other goal I have in mind is to be able to solve Chu et al.'s shortcomings with respect to go routes and comeback routes. If I can differentiate between the two routes in question, I will consider that a moral victory even if the clustering of the routes doesn't go the way that I like. With all that said, let's take a look at how my analysis went.

Pre-Processing

The initial dataset that I used was the week 1 data, containing 986,022 samples, or data captures, with the 19 original features. From there, I narrowed my features down to the 12 aforementioned key features, as well as cutting out the samples that did not have a route label. This gave me the timestamps of only players that were running routes. This narrowed my dataset down to a 273,314 x 12 dataset. From there, I cycled through the data and extracted what I defined as routes.

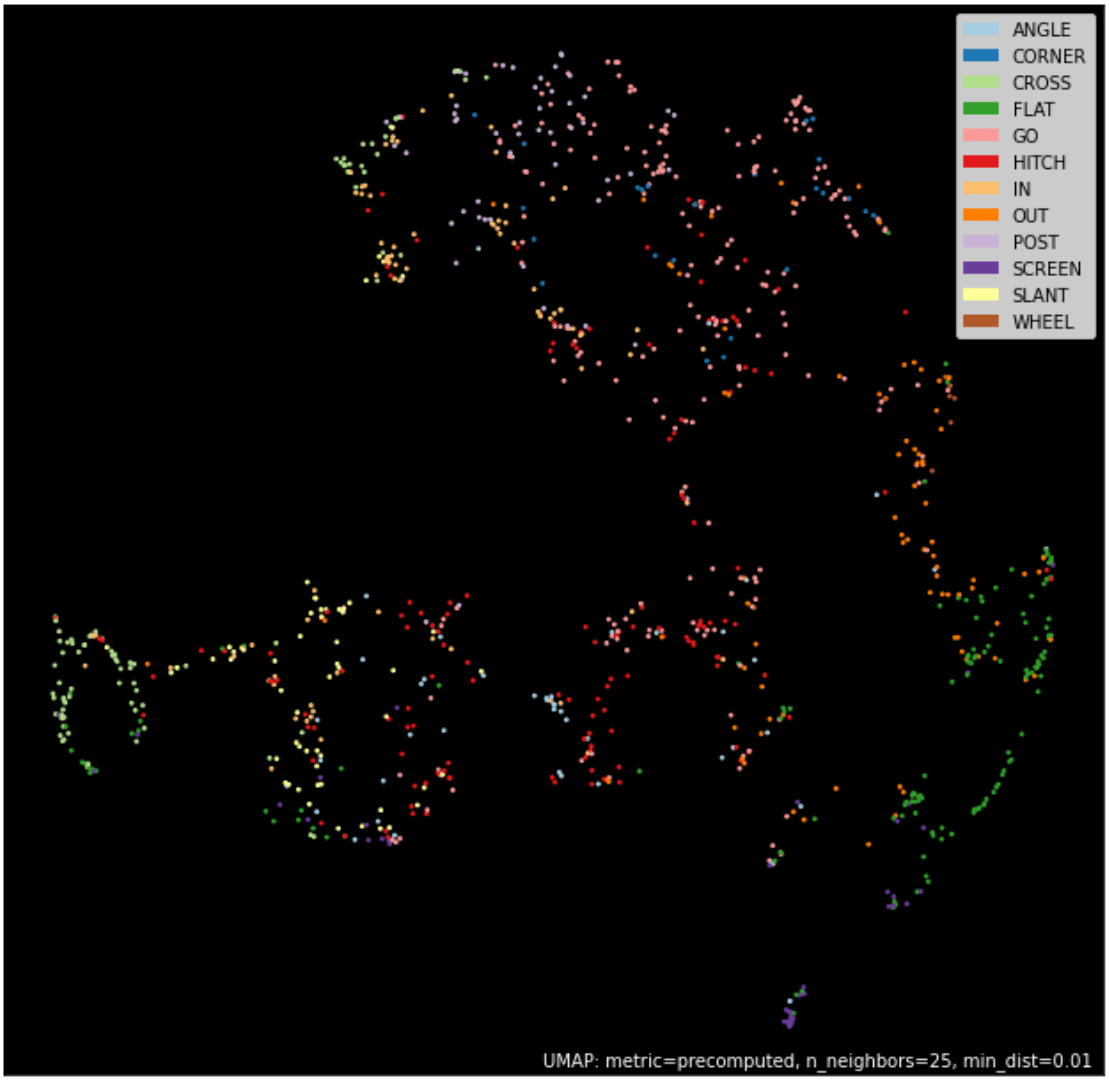
Throughout this process of extracting routes, I had to make some choices that I believe affected my results in some capacity, but I will touch on that more later. One of these choices was to only work with plays that had completed passes, so I specifically captured routes from the "ball_snap" to "pass_outcome_complete" events. While isolating routes, for the sake of computational time, I chose to only consider every 5th data point within a route. This preserved the general shape of the route that was run as well as making my analysis cost less computationally. After isolating 2,554 routes, my next task was to standardize my data.

As the tracking data only contained positional coordinates, I had to account for the location on the field, the direction of the play, and which side of the ball the route was starting on in order for any meaningful analysis to be done. For instance, I want two in routes starting on different sides of the ball (meaning the cut direction will be different) to count as the same route. This is only possible if the positional data is mirrored. Similar problems occur when accounting for play direction.

Once standardized, I decided to choose the first 1,000 routes to perform dynamic time warping on. Doing all 2,554 took me way too long, but limiting the number of routes I had to compare helped me to calculate the 1,000 x 1,000 dissimilarity matrix in around 30 minutes. This computation included calculating the dynamic time warping distance between every pair of x coordinates, doing the same for the y coordinates, and then summing the two distances together. It helped a great deal computationally that this matrix was symmetric.

Data Analysis: UMAP

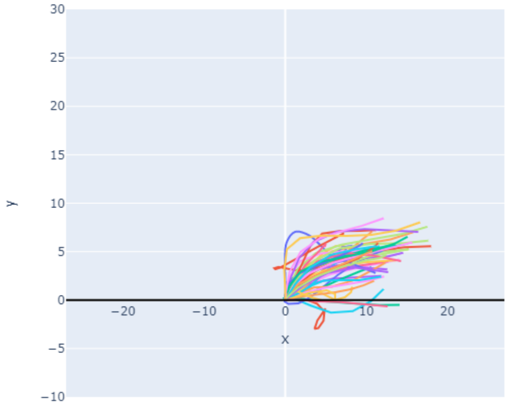
Deciding which methods to use for analysis was solely based on the availability of this dissimilarity matrix that I calculated. Since the coordinate arrays for each route varied in length, I would not be able to justifiably compare them as vectors. I wanted to see what sort of embedding this dissimilarity matrix could provide, so I had to choose a method that accepted dissimilarity matrices as input. I eventually landed on UMAP and decided to use the labels that the data set originally provided for the routes run for visualization despite not knowing how the data was originally labelled. I felt that whether the labels were assigned by someone watching the film of the original play or whether it was determined from an algorithm didn't matter since UMAP doesn't rely on labels in its embedding process.

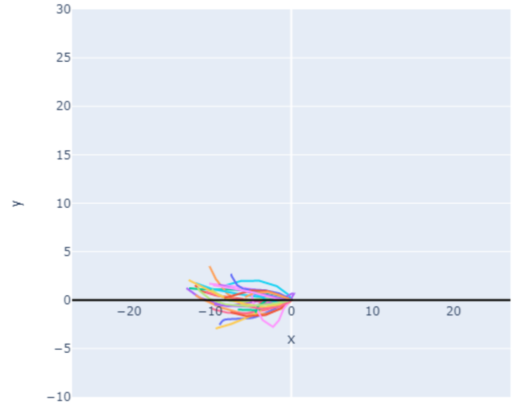


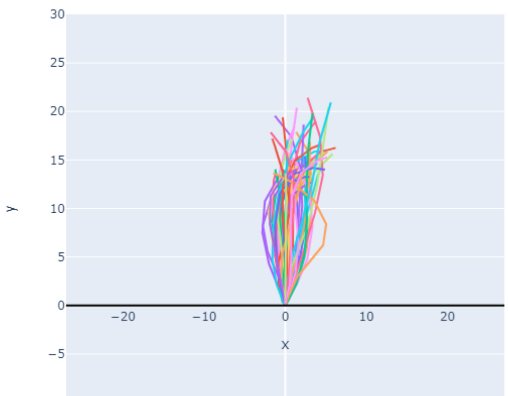
Note, in my analysis of this embedding, I am going to assume the labels of the routes are correct. Looking at the embedding, it is clear that there is very little separation between different routes. Some grouped fairly well together, such as screens, flats, and crossing routes. Outside of that, there are many routes that are dispersed within each other. For instance, hitch routes (otherwise known as comeback routes), got spread throughout the embedding. I suspect that this, along with most of the problems in the analysis, is due to issues with using dynamic time warping and/or the data itself. I will elaborate on my methodology's shortcomings later in this paper.

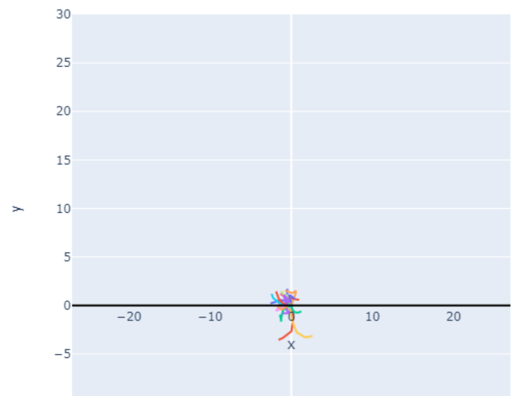
Data Analysis: Spectral Clustering

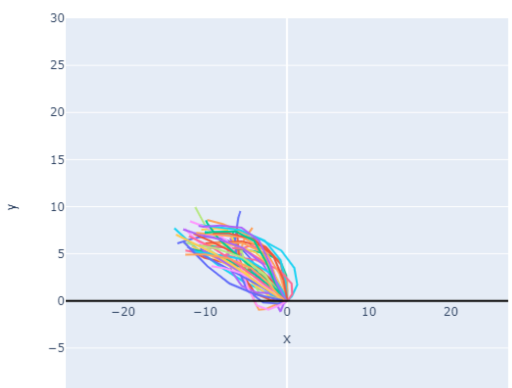
Next, we will take a look at the spectral clustering of these routes. Specifically, I attempted to capture 24 clusters, assuming 2 clusters for each of the 12 different labels that were provided in this set. This assumption is made because similar routes may in practice be run at different depths of the field. I selected spectral clustering due to its reliance on a dissimilarity matrix. When plotted together, the routes within the clusters yield some promising results, as well as some discouraging ones. Let's start with some of the good clusters.

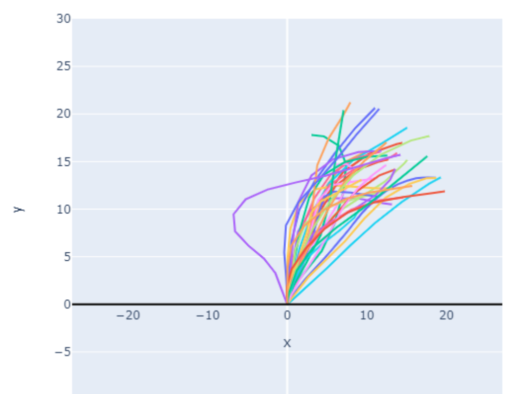

y
x


y
x

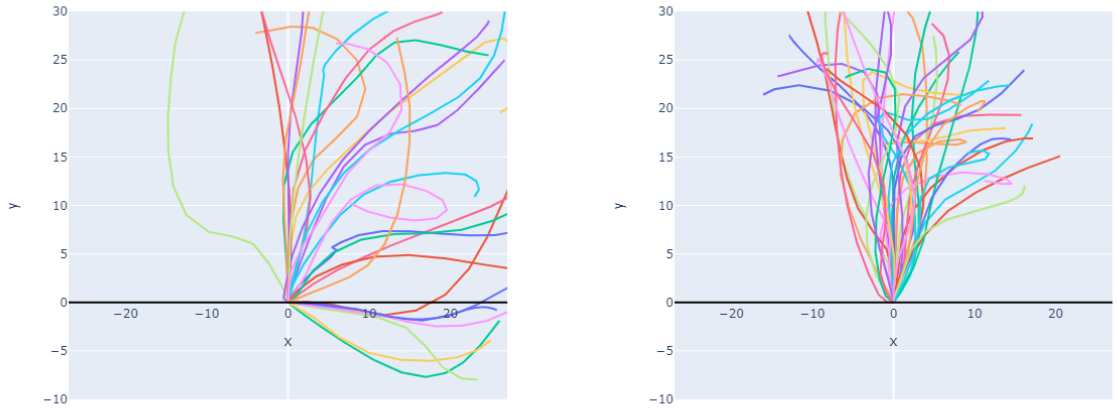

y
x


y
x


y
x


y
x

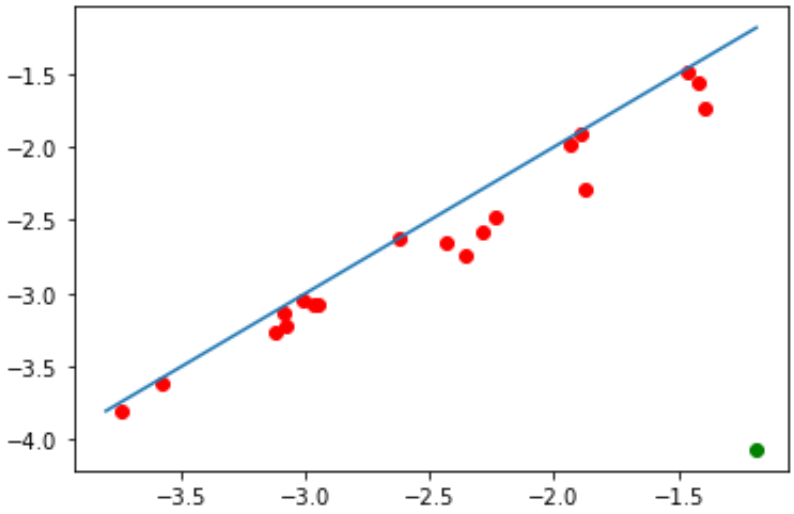
As you can see, some of these clusters, although cluttered, seem to capture the shape of the routes fairly well. The first cluster looks like a bunch of crossing routes, the third clustered screens together, and the final cluster looks like in routes. Within each cluster, there are some routes that seemingly don't belong, but for the most part, they look pretty well grouped. Sadly, this is not the case for all clusters.



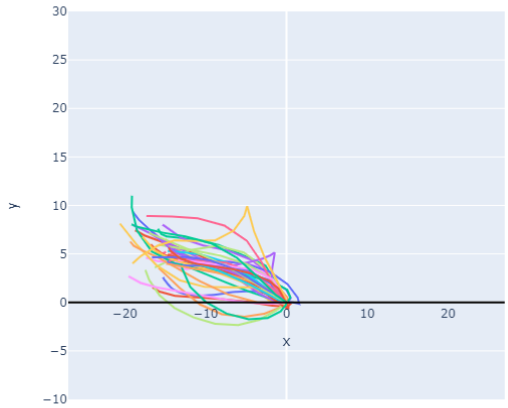
It would be nice if I could chalk up these odd clusters to routes that I could consider outliers, however, there are clearly some normal routes contained in these clusters. As to why these routes are clustered this way, I am not entirely sure. I have a couple of theories, but more on that later.

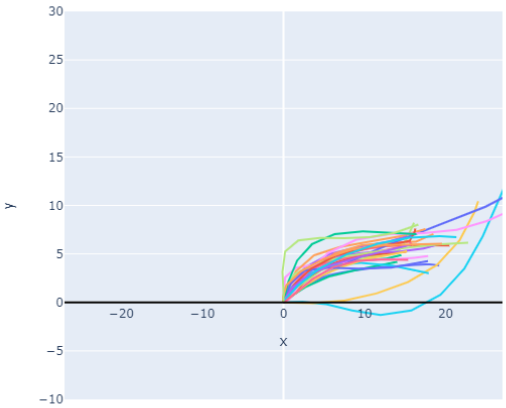
Data Analysis: ToMATo Clustering

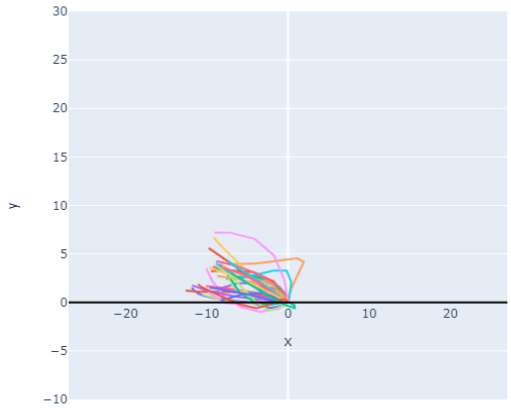
Just to make sure that it isn't an issue with using spectral clustering, I also performed ToMATo clustering on the same dissimilarity matrix to see if I could yield better results. It is important to note that ToMATo Clustering determines how many clusters are created using persistence. According to the results, there were 19 clusters obtained. Below is the persistence diagram, each point representing a cluster.

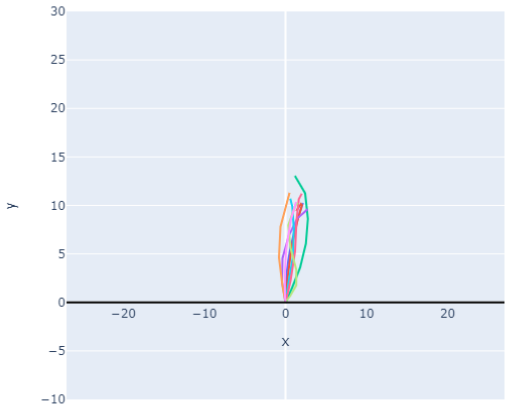


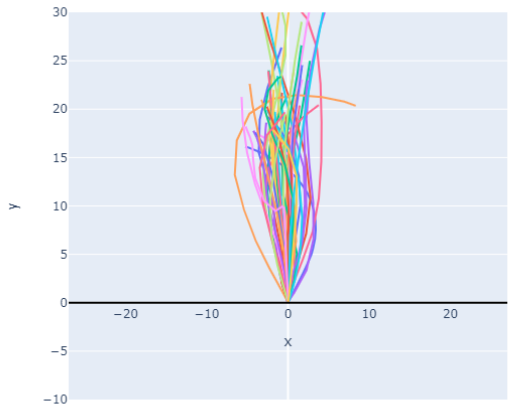
That said, let’s take a look at some of the good clusters first.

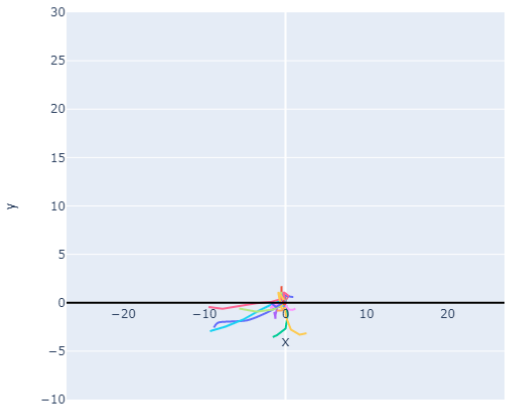




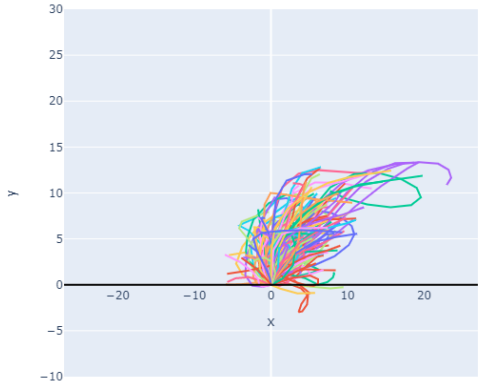


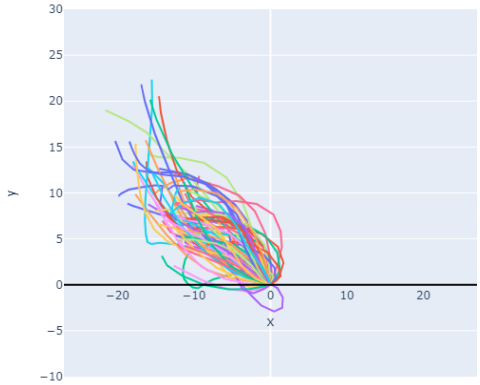


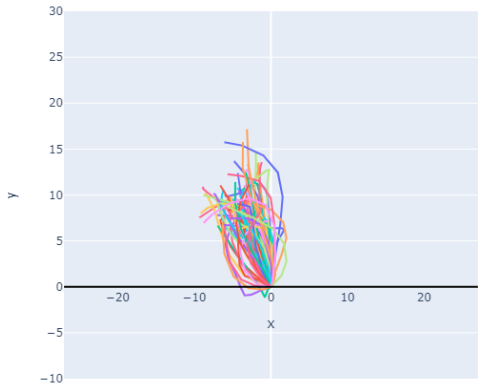


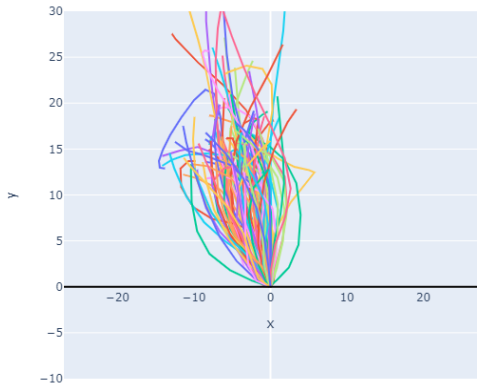


As you can see, some of these clusters contain very similar routes, with a few routes as exceptions. Despite this, these clusters look very empty compared to the spectral clustering. So what happened to these routes? Well, let's take a look at some of the more concerning clusters.









Specifically, the first and last of the problematic clusters looks like they contain a large amount of fairly dissimilar routes. The rest of them, while they have routes of somewhat similar shape, appear to have multiple styles of routes within the clusters.

Observations

So it seems that neither of the clustering methods worked very well. Why might this be? Let's start by discussing the use of dynamic time warping as a dissimilarity metric for this data. While in theory, it seemed that it would prove to be a useful tool, I think dynamic time warping failed to capture the nuance of the changes in similar, but different, routes. For example, we see a lot of routes that utilize similar field positions clustered together, despite the path that they are taking being pretty different. Due to the nature of dynamic time warping and the dissimilarity calculations we made, I don't believe there was a harsh enough penalty for our methods to actually capture the nuance of similar, but different NFL routes.

I also believe that some of the larger problem lies in the data and how I am able to work with it. Due to the structure of the data, I had to make a lot of assumptions and compromises that I assume led to skewed analyses and results.

The first assumption that I made was the initial cutoff for where a route ends. With a multitude of play-ending events to work from, I had to limit myself to working with plays that only resulted in a caught pass because I couldn't guarantee I wasn't cutting a route off too soon. That said, just because a play resulted in a caught pass does not mean that every eligible receiver completed their route. The data set does not allow me to differentiate which players caught the pass, so I have no way to only isolate truly completed routes. This can lead to certain routes that should be different being put in the same cluster. For instance, in reality, a slant route will look very similar to a crossing route that has been cut off early.

The second assumption that I made was that only taking every fifth data point would not change the results in a meaningful way. While this method still captures the shape of the route fairly well, when paired with dynamic time warping, this may skew our dissimilarity values.

My last assumption is that defense wouldn't have a large impact on the position of some of these routes. Due to a defensive player trying to break up the play, a route ran during a game might look different than the way the play was written up on paper. I had hoped that this wouldn't reflect too much in my analysis, but it is potentially another factor that could have skewed my analysis.

Future Work

In the future, I believe there are a couple of things that I could improve upon. First, I could consider how my dissimilarity metric is calculated. Are x/y coordinates truly the best thing to compare using dynamic time warping? Would polar coordinates be better? Could I do something besides just adding up the two distances calculated to get the dissimilarity between two routes? Could I incorporate speed or acceleration into my analyses somehow? Lastly, is there a better distance algorithm to use in place of dynamic time warping? All of the answers to these questions may result in a better attempt at clustering routes using this data, despite the problems within the data set itself.

Conclusion

While my analyses did not yield the results that I wanted, I believe that this was a solid start at attempting to cluster NFL routes using positional tracking data. Dynamic time warping provided to be a useful tool, and despite its downfalls, it is what made this approach possible. That said, all of my results were not bad. In fact, most clusters had a lot of similar routes grouped together. This gives me hope that I can expand upon this method and improve it in the future, and I look forward to doing so.

References:

1) Gholizadeh, S., Savle, K., Seyeditabari, A., & Zadrozny, W. (2020). Topological Data Analysis in Text Classification: Extracting Features with Additive Information.
2) "NFL Big Data Bowl 2021." *Kaggle*, https://www.kaggle.com/c/nfl-big-data-bowl-2021/data?select=week1.csv.
3) "NFL Next Gen Stats." *NFL Football Operations*, https://operations.nfl.com/gameday/technology/nfl-next-gen-stats/.
4) Zhang, Jeremy. "Dynamic Time Warping." *Medium*, Towards Data Science, 14 Feb. 2020, https://towardsdatascience.com/dynamic-time-warping-3933f25fcdd.