Using Image Transformations to Learn Network Structure

Brayan Ortiz[1*] and Amitabh Sinha[1†]

[1*]Modeling and Optimization, Amazon.com, Seattle, WA, USA.

*Corresponding author(s). E-mail(s): brayanvo@amazon.com;
Contributing authors: amitabsi@amazon.com;
[†]These authors contributed equally to this work.

Abstract

Many learning tasks require observing a sequence of images and making a decision. In a transportation problem of designing and planning for shipping boxes between nodes, we show how to treat the network of nodes and the flows between them as images. These images have useful structural information that can be statistically summarized. Using image compression techniques, we reduce an image down to a set of numbers that contain interpretable geographic information that we call *geographic signatures*. Using geographic signatures, we learn network structure that can be utilized to recommend future network connectivity. We develop a Bayesian reinforcement algorithm that takes advantage of statistically summarized network information as priors and user-decisions to reinforce an agent's probabilistic decision.

Keywords: Machine Learning, reinforcement learning, Bayesian, logistics

1 Introduction
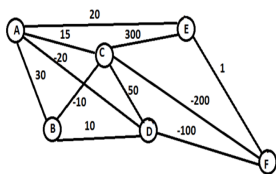
Fig. 1: Mock topology and cost structure.

In transportation and logistics companies, decisions have to be made frequently by operators on which warehouses or nodes need to be connected, i.e. the travelling salesperson problem (TSP). This problem has constraints and structure. We can abstract this down to a simpler scenario as in Figure 1. We want to ship a package from A to F with minimal cost, where cost is shown on the edges. Nodes C, D, and E are relatively well-connected to F, i.e. we are "positively rewarded" for taking $C \to F$ and $D \to F$ (cost is negative as in savings), while $E \to F$ is only \$1. Cost can be the shipping cost of each box or the cost of ordering trucks. Unfortunately, in practice, such a topology and structure of costs becomes obsolete either because of stochasticity or evolution: new nodes are added (new node G either within the graph or outside of it), new connections ($A \to F$ or $B \to E$ directly, $A \to G$, etc.), and most importantly the cost structure is changing based on stochastic demand and pricing.

This paper focuses on the decisions made by an operator choosing short-term network activity. In a short enough window, changes within a network are likely small, but identifying consistent connections and recommending new connections based on network activity is valuable. Airlines, for

example, change connection structure only every so often. Our goal is to define a mathematical representation of a transportation network that allows experimentation of new nodes or connections by an operator that is making short-term decisions on which connections should be opened.

Ideally, to choose connections that will be active (have boxes shipping), we can forecast the number of boxes flowing within all connections for a given network while respecting network or geographic constraints, such as node capacities or transit times. The traditional approach to network design and planning in operations is to consider global optimization techniques that solve for connectivity and flows while constrained for asset capacities [6, 9]. This global approach is useful for strategic or long-term decisions. Batch or campaign style planning is well-studied in industrial processes, but in transportation it could only be effective provided standing decision mechanisms are aligned, such as simultaneous launch of buildings, etc. [3, 7, 8, 11]. Overall, when we approach this problem by embedding unique node and time specific information, then the mathematical model does not generalize beyond that network state. If we can summarize network activity at some aggregate level while capturing network dependencies, then it would be possible to estimate the cost of new networks with added nodes and connections.

For the TSP problem in general as represented by structural graphs, there are computationally expensive methods that leverage learned abstractions of historical graphs via deep neural networks to explore unseen instances of graph activity [2, 10]. However, it would be difficult to experiment with such a method if the user has geographical properties in mind, such as location of a connection relative to a highly active node. In this paper, we show that these flows can be represented as images unique to each node and time point. Using compression techniques to reduce the dimension of the images, we can summarize the position and relative importance of each node at a particular time. We call these geography-specific compression summaries *geographic signatures* (Section 2).

Using low-dimensional representations of graphs has been effective in network analysis tasks [4, 5, 14]. We want to use the geographic signatures to capture historical network structure and infer relative importance between origin

nodes and a destination (a rank), while also allowing experimentation with future network structure. Using geographic signatures along with live operator-feedback, we show that we can learn the relationship between historical network activity and node pair flow/connection. Although rank features have been used in TSP problems [15], it has not been treated as a target that is updated and reinforced. Online learning approaches via low-dimensional representations have been looked at for social network learning [12]. Here, we develop methodology to extend the idea to operations bound by geography. In Section 3, we propose a framework for using the geographic signatures to train a probabilistic model that ranks potential future connections and learns based on actual user decisions. We conclude with future work.

2 Summarizing Transportation Network Activity Using Geographic Signatures

In this paper, we describe our work on the transportation problem of an e-commerce company, referred to here as XYZ. At XYZ, inventory is stored within warehouses across the United States (Fulfillment Centers or FCs for short). From within the FC, a purchased item from the website is picked, packed and moved onto a truck to be moved through the transportation network. A package at an FC can move to a sortation node where a package will get sorted and put onto a truck with other packages going to a common delivery node. Alternatively, a package can also move directly to a delivery node without going through a sortation node. There are other variations possible, but the transportation of a package between the origin warehouse and a delivery node is what is known commonly in the industry as the 'middle mile.'

A central problem in middle-mile network design is predicting origin-destination (OD) flows: How many packages will flow from each origin warehouse to each Zip3/Zip5/facility/etc on a given date in the future? There are many possible approaches to this problem across the industry: optimization via mixed integer or linear programming or forecasting via statistical time series models are two major examples. However, in the optimization case, it is easy for the number of variables to blow up for a problem that has a large

network or a large horizon of time, so simplified problems are typically considered that preserve structure and process as much as computation allows. While in the statistical case, it can be difficult to effectively capture geography and other process constraints, but time and uncertainty can be modeled readily. We want to focus on capturing geography in a problem.

In this paper, we introduce a mathematical construct for bringing in geographical information into modeling, by using a concept we call geosigs (abbreviation for "geographic signatures"). This concept's construction draws from the field of lossy image compression [1]. After motivating the problem in Section 2.1, we define the compression summary for this problem as geographic signatures (2.2). We demonstrate the value of geographic signatures by building a simple OD flow prediction model for the United States and observe how the geographic infromation can be utilized for prediction and experimentation.

2.1 Motivating Problem

Consider the situation in Figure 2, where we have two origins at equal distance from destination Zip_0, where Zip0 is some zip-code of customers. The question we are trying to answer is the following: "How much volume flows from each FC to Zip0?" A purely distance-based model will assign equal volumes to both FCs, because they are equidistant from Zip0.
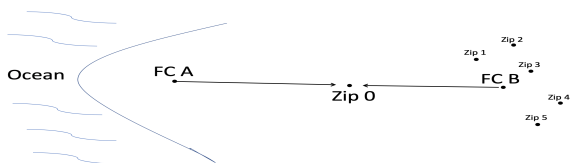
Fig. 2: Mock topology.

However, the picture shows some potentially useful geographic information. FC_A is surrounded by an ocean, with no population centers around it. FC_B, on the other hand, is surrounded by population centers. One would therefore expect that FC_B sends much of its volume to its nearby population centers, and has very little to send to Zip_0, while FC_A fulfills a disproportionately higher fraction of Zip_0's demand. It is possible to capture this implicitly via capacity constraints, but our premise is that we should not need to go to force the constraint. We should enable our models to capture what is easily visible to the human eye: the different geographic embeddings of the two FCs must result in different predicted flows to Zip_0. The rest of this section describes how we propose doing this, and what our test results show.

Throughout, we use historical flow data from XYZ that has been masked. All analyses are done using Python on an i7 2.5 Ghz machine with 16 GB RAM.

2.2 Geographic Signatures Definition

We begin by providing the formal definition of how we compute compression summaries in this setting, followed by detailed discussion of each step. For notation's sake, let $ZipN$ denote the first N digits of a zip code, e.g. for 98109-4385, Zip3=981, Zip5=98109, and Zip9=98109-4385. Also, we abbreviate latitude ('lat') and longitude ('lon').

Let G be a collection of nodes, such as the collection of all Zip3s, or the collection of all FCs. Let x be a single node, which may or may not belong to G. For every node, we are given the latitude, longitude, and a measure f. The measure may be just a constant 1 in the simplest case; or it may be something more appropriate for the context: e.g. if G is the collection of all FCs, then f may be the capacity of each FC, or if G is the collection of all Zip3s, it may be the "purchasing power" of each Zip3.

The compression summary, $CS(\cdot, \cdot)$ of $x \in G$ is defined as:

$$CS(x, g) = PrincipalComponents(FFT(Polar(x, G))).$$

$Polar(x, G)$: This is a matrix that represents (G, f), as viewed from x, in polar coordinates, with some discretization. Consider a node v, defined by $(lat(v), lon(v), f(v))$. The polar representation of its location with respect to x is given by $Polar(x, G) = (r, \theta)(x, v)$, where $r = \sqrt{(lat(v) - lat(x))^2 + (lon(v) - lon(x))^2}$ and $\theta = \tan^{-1}\left(\frac{lat(v) - lat(x)}{lon(v) - lon(x)}\right)$, with appropriate manipulation of the sign of θ so that starting from East and rotating counter-clockwise, θ goes from 0 to 2π (in radians).

(a) Los Angeles   (b) Houston
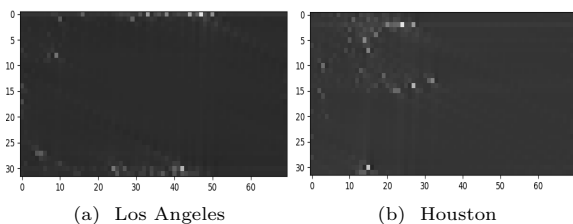
Fig. 3: Bitmaps translating flows from two FCs, in Los Angeles and Houston, to a collection of Zip3s for a particular day.

We choose the step sizes of r and θ. For θ, suppose we can choose the four cardinal directions, steps of $\frac{\pi}{2}$, or smaller steps of $\frac{\pi}{16}$. For r, choosing $r = 1$ on a map in latitude and longitude is similar to choosing $r' \approx 69$ miles. We might choose r as the expected length of a truck route to reflect the actual segmentation of a transportation network. For any chosen set of step sizes, we sum the f values for all nodes in G that fall into a common neighborhood of r and θ. In general, applying polar coordinates with discretization yields $P \in \mathbb{R}^{d_r \times d_\theta}$, where d_r and d_θ denote the number of discrete points in the range of r and θ, respectively.

We can visualize this polar matrix as a bitmap. Two such bitmaps are shown below for two FCs in the XYZ transportation network (Figure 3), one in Los Angeles and the other in Houston (each FC can be treated as the top left corner). In each, θ is on the Y-axis, with $Y = 0$ representing East, and as we go down the Y-axis, going to North, West, South, and back East. The X-axis represents distance. Here, G is the set of all Zip3s, and f is the total volume at each Zip3. We see that a lot of the far distances (the right half of the images) are empty and this is mainly because an FC in LA may not typically be the origin of packages going into the eastern corners of the US. A little more than just the bottom half of the LA picture is also dark (zero measure), which is explained by the presence of an ocean Northwest, West, and Southwest of LA. Houston has some destinations West and close, but not far.

$FFT(Polar(x, G))$: Next we apply a Fast Fourier Transform to the matrix $Polar(x, G)$ yielding a matrix $F \in \mathbb{C}^{d_r \times d_\theta}$. FFT is also an invertible matrix transformation, from the real space to the complex space with the same dimensions. It decomposes the original image using simpler basis

(a) $mask_{max} = 1$ (b) $mask_{max} = 3$
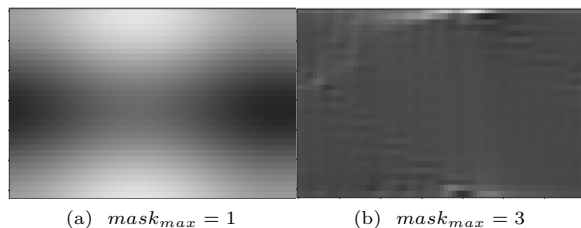
Fig. 4: For Los Angeles, the image after applying a mask on F and inverting onto the magnitude spectrum.

functions. A consequence is information "concentration": FFT (like other such transforms) forces most of the signal to be at matrix positions closest to (0,0) (as we have defined it).

$PrincipalComponents(FFT(Polar(x, G)))$: Finally, taking advantage of the information concentration done in the FFT step, we select the entries (i, j) of F such that $i+j \leq mask_{max}$, which we call the principal components of the matrix for ease of exposition. In one of the results displayed, we only selected 3 matrix ($mask_{max} = 1$) elements to summarize the position between each OD pair: (0,0), (0,1) and (1,0). This is where we accomplish compression: the entire picture is represented by 3 complex numbers, or equivalently, 6 real-valued numbers. In fact, the imaginary component of the (0,0) element of the FFT is always 0, so we're really only using 5 numbers to represent the entire picture. These 5 numbers define the compression summary, or $CS(x, G)$.

In Figure 4, we show two pictures which are inverses of the LA picture displayed on the previous page, with $mask_{max}$ equal to 1 and 3 respectively. Observe that neither picture is exactly equal to the LA picture shown in FIgure 3, e.g. "lossy" image compression. Selecting more principal components (higher $mask_{max}$) results in an inverted image that is close to the original: selecting a smaller value of $mask_{max}$ results in greater information loss. In that sense, $mask_{max}$ may be thought of as a "tuning parameter."

These summaries do not reside on the domain of the original image, which is why in the previous example we had to invert the image. In the original images on the real domain of flows, each row/column pair correspond to an angle/distance historical flow. Once we apply the mask on the coefficients and transform, we get instead a smoothing of flows on a magnitude spectrum. In signal processing, each complex coefficient $\alpha^* = (r, c)$ can

be transformed such that the power, *power*, can be expressed as $power = (r^2 + c^2)^{(1/2)}$. Here, that power is interpreted as instantaneous flow or flow intensity.

When we use the first three upper left corner coefficients using $mask_{max} = 1$, then we get the average flow intensity at the origin (0,0), 1 degree away from the origin (1,0), and 1 turn or angular step from the origin (0,1). For the LA image, this might be useful for northern flow activity, but it will not capture southern flow activity. With a mask of 3, we increase the number of coefficient and flow intensity estimates to 6 (so 6 potential features for a regression model). However, we learn nothing about where flows occur, only about the lack of flows close to the FC, which is useful information for problems dependent on local network activity. Ideally, the level of discretization is fully utilized, since this is the most preserved level of information. In Figure 4, we observe how discretization can be interpreted on the magnitude spectrum (inverted image) as a smoothing of the observed flows. We propose a summary of the magnitude spectrum that fully utilizes the discretization in a meaningful way for many network problems.

The geographic signature, $Geosig(\cdot, \cdot)$ of $x \in G$ is defined as:

$$Geosig(x, g) = MagnitudeSpectrum(CS(x, G)).$$

We use this transformation to summarize network structure and reduce dimensionality of the feature space. We propose to discretize the angles of the network into 4 angles: $(0, \pi/2), (\pi/2, \pi), (\pi, 3\pi/2)$, and $(3\pi/2, 2\pi)$. This discretization follows the 4 directions: NE, NW, SW, and SE (in that order). Next, we propose distances of 100 miles for simplicity. Hence, we discretize each network flow image into 4 rows and 17 columns. We propose using a semi-coarse mask ($mask_{max} = 2$ to be discussed) and summarizing the $4 * 17 = 68$ geosigs by the maximum smoothed flow along each row or direction. Furthermore, each max flow is paired with the position of the max, i.e. the number of 100 mile segments that it takes to get to that maximum flow intensity. In this way, we preserve specific and interpretable information, namely, how far a Zip3 can be from a particular FC to be fulfilled.

For other problems, alternatives include using the first non-zero flow intensity (requires a fine mask) or using the first flow intensity above an interesting threshold (say 3000 packages). The geographic signatures or geosigs can be tailored to the problem. In what follows, we demonstrate the utility of the first proposed summary of geosigs, the max flow pairs.

2.3 Explaining Historical OD Flow with Geographic Signatures

We now demonstrate the value of compression summaries and geosigs by building a simple OD flow prediction model.

The problem statement is the following: for a single selected date in the future, predict exactly how many packages will flow from each FC to each Zip3. If items were in inventory at the closest facility to a customer always, then this would not be such a difficult problem. Instead, we have to expect that many facilities will help to fulfill the demand from a particular destination, either because of rarely ordered items or items that can only be stored in special FCs. In this section, we estimate a linear regression model where the dependent variable is the number of packages in each $FC \rightarrow Zip3$ pair, and we use several independent variables explained next. Models considered are trained on a single day: this is day 221 in the year 2018, which is Thursday, August 9, 2018 (18,635 pairs). We test the models against day 249 of 2018, which is Thursday, September 5, 2018 (21,042 pairs). The flow data used in these studies has a controlled amount of noise to retain a particular degree of the original signal, while aliasing the true values of the confidential data. We evaluate the utility of the geosigs as explanatory and predictive variables using simple regression models evaluated via mean actual percentage error (MAPE) and adjusted R^2. At the scale R^2 is measured, we introduced at least 80% of the variation in the noisy data, so $R^2 > 0.10$ will be desirable.

The linear regression models use the following independent variables:

- ln_lld: Natural logarithm of the Euclidean lat-lon distance between the FC and the Zip3. Included in all models.

Model A includes 14 geosig variables (dDI_01, dDI_10, dDR_00, dDR_01, dDR_10, oOI_01, oOI_10, oOR_01, oOR_10, oDI_01, oDI_10, oDR_01, oDR_10, and oDR_00), defined according to the following convention:

- oDR_00: This computes the geosig of each origin FC, with respect to the matrix of all FCs, and extracts the real part at matrix location (0,0).
- dDI_01: Geosig of each Zip3, with respect to the matrix of all Zip3s, imaginary component, matrix location (0,1).

Model B uses magnitude or power of the first coordinate (interpreted as average flow intensity) of each domain, denoted as oO_00, oD_00, and dD_00. Model C and D use 4 max flow distance pair values (8 values per OD pair), with the following convention:

- oD_0r_summary_max: This computes the max intensity along the 0th direction $(0, \pi/2)$ for $FC \to Zip3$.
- oO_0r_summary_max: This computes the max intensity along the 0th direction $(0, \pi/2)$ for $FC \to FC$.
- dD_0r_summary_max: This computes the max intensity along the 0th direction $(0, \pi/2)$ for $Zip3 \to Zip3$.
- oD_0r_summary_max_r: This computes the distance, r, at which the max intensity occurs along the 0th direction $(0, \pi/2)$ for $FC \to Zip3$.

Model C uses only the max flow (4 values per OD pair). These are all computed with the measure f being the total flow on the training date: that is, for the matrix D, the measure f of each Zip3 is the total number of packages delivered to that Zip3, and likewise for FCs in O. We use three sets of geosigs: origin domain (o,O),origin-destination domain (o,D), and destination domain (d,D). For each domain set, we extract the real components at matrix indices (0,0), (0,1) and (1,0), and the imaginary components at (0,1) and (1,0).

In Table 1, we present the results of fitting linear models using geosigs. The best predictor of day 249 was Model B with 57% MAPE, although it had a low R^2. Using the average flow intensity along each domain resulted in a low variance, but higher bias estimator. We see that raw geosigs can explain about 19% of the variation in the training data (Model A), but summarizing the geosigs not only improves test MAPE (Model C, Model D), but also provides a more generalizable model (Model D with highest R^2).

Overall, this shows the improvements we gain by tailoring geographic signatures for a flow problem. By using geographic signatures, we reduced

| Model	Adjusted R^2	Train-MAPE	Test-MAPE
A	0.19	0.87	0.79
B	0.06	0.87	0.57
C	0.13	0.85	0.78
D	0.31	0.77	0.71

Table 1: Adjusted R^2 values for linear models fit on flow data using features engineered using geographic summaries.

the dimensionality of the network, preserved the signal relevant to the relationship between geography and OD flow, and retained features that are interpretable. We next show how to use that interpretability to learn fundamental network properties that allow us to consider hypothetical networks.

2.4 Hypothetical Networks

All of the measures that we are considering as belonging to a network or topology are intrinsically dependent on each other. This means even when we summarize the information into geographic signatures there is correlation among the features. We want to capture this collinearity to have an understanding of network effects that generalizes to other within-network scenarios. We fit a gradient boosted random forest to capture some of that dependence and demonstrate how the geosigs allow us to play out basic scenarios such as arc additions or new facilities.

Using the features in Model D (distance and the max flow pairs), a gradient boosted regression forest was trained (`sklearn` with 1000 boosting iterations, otherwise default settings) and had a test MAPE of 63.6%. When observing the flow tendencies of the network between the strongest pairs of FCs and destinations in the NE direction ('oD_0r_summary_max'), the pairs with an average flow intensity less than 800 boxes (about 90% of arcs based on Figure 5) tended to have a similar overall flow that was smaller than the flow expected from pairs with an average flow intensity greater than 800 boxes. These NE-directed max pairs tended to occur quite far (close to 60% occurred at more than 1500 miles) and those large distances tended to be associated with smaller flows overall as can be seen in the PDP for the distance of the max pair, 'oD_0r_summary_max_r'. Overall, this suggests a network that has a particular relationship with the NE direction: it has FCs that serve it thick arcs from about 1500 miles, but
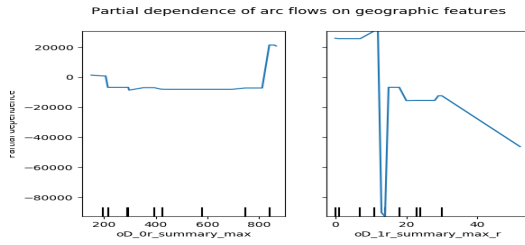
Fig. 5: PDPs look at a variable of interest across a range. At each value, the model is evaluated for all observations of the other model inputs, and the output is then averaged. A negative (positive) value indicates that the feature value may be associated with smaller (greater) flows compared to the rest of the range averaged across the other features. The ticks mark quantiles.

for the most part NE directed arcs are not that thick.

Using the partial dependence values directly, we can estimate the contribution of a particular type of activity to a network. In the example given, we can estimate the total change in flow to a network relative to the activity of NE-moving packages. Under the network represented by the data sampled and transformed, we can expect 8931 fewer boxes moving NE than the average of the other directions. NW moving boxes are the minority averaging about 10,253 fewer boxes than the network average. The most active overall direction under this summary is the SW set of boxes at about 6491 above average.

Suppose we place a hub in the middle between Los Angeles and New York at about 800 miles from Los Angeles, so that we can reduce the overall average arc length of the network, call it HUB. Using the PDPs, we estimate changes in flow across the network. We want HUB to play an important role for the SW region, so we want network information to reflect this new activity. We generate data such that all origins SW of HUB will have 'oD_0r_summary_max_r' equal to the number of segments between the origin and HUB. 'oD_0r_summary_max' might tend to be high around 800 boxes. Other than this behavior change in the network for the SW region, HUB is not going to impact any other origins. This implies that getting the average value for all other geosigs will suffice.

In a small experiment, we replace 25% of the test data to reflect the SW region's new HUB

and randomly generate distance and two of the geosigs as Gaussian random variables centered at the means of interest. Partial dependence on generated data quantifies how the network tends to be associated with that activity summary. Using the geosigs for the historical network, we can expect 17,382 fewer packages going NE with the addition of HUB, but 12,226 more boxes to be moving SW. When one area of the network experiences more activity, the learned network associates that with a decrease in activity for other areas. This experiment highlights the dependence that can be captured using geographic summaries. Under this experiment, we would need to know specific arc cost information to estimate whether this HUB would lead to overall reduction in network costs or other important metrics. In the next section, we consider the problem of network activity at a larger scale and utilize the geographic signatures to capture network structure.

3 Lane Recommendations Reinforced by Users

We define the short term problem of deciding arcs for the next time period based on last time period as a cascading bandit problem [13]. We train a probabilistic model, a cascade model, that governs the arcs that will be recommended to a business operator. This probabilistic model can be informed by our prior knowledge of the network as encoded by the geographic signatures. In Section 3.1, we rank arcs particular to a destination based on cost of a package on that arc derived from a regression model that takes as features geographic signatures among others. We rank by cost because we want to minimize the overall cost of the network. Using these ranks as priors, we propose a Thompson sampling algorithm to learn the probabilistic model that governs arc recommendations (Section 3.2). Using Bayesian techniques for reinforcement learning, we show in Section 3.3 that it does not take many iterations to learn important network structure under a consistent network topology. We also discuss user-driven agent-assisted experiments to summarize new topologies.

3.1 Ranking Arcs Using Geographic Signatures

Using historical arc costs (based on truck costs) and volumes, we want to predict future arc connections. To do this, we will rank the arcs based on cost and rank low cost arcs high. First, we must estimate cost. For demonstration, we propose a regression model for weekly data with the following data structure. Let FC_i and D_j denote the ith origin FC and the jth destination node. D_j can be a delivery node or a hub. We define $c_{i,j,t}$ as the average historical cost per package and $n_{i,j,t}$ as the number of packages or flow on arc $FC_i \to D_j$ at time t. Additional measures we consider in this demonstration include: FC_i latitude and longitude, lat_i and lon_i; the distance between FC_i and D_j, $d_{i,j}$; and a binary indicator $direct_{i,j} = 1$ when D_j is a delivery node (otherwise, 0).

We need to be able to capture network topology of the United States. We want to know when a facility serves locally more than across the US or when there is a body of water or a mountain that explains why a facility does not serve a destination. However, we want to embed this information as simply as possible. Using flow and location $(f_{i,j}, lat_i, lon_i, lat_j, lon_j)$, we define geographic signatures as done in Section 2.2. In particular, we use the max flow pair summaries introduced in Section 2.3, where $g_{i,j} = geosigs(f_{i,j}, lat_i, lon_i, lat_j, lon_j) \in \mathbb{R}^8$. Let $x_{i,j,t} = [t, direct_{i,j}, f_{i,j}, d_{i,j}, lat_i, lon_i, g_{i,j}]$. We aim to estimate the expected cost of an arc at time t,

$$\hat{c}_{i,j,t} = \mathbb{E}\left[c_{i,j,t} \mid x_{i,j,t}\right] = \hat{f}(x_{i,j,t})$$

, where $\hat{f}$ can be estimated using random forest regression techniques.

Using 52 weeks of historical data (spanning October 2018 to October 2019), we fit daily $\hat{f}_{i,j,t}$ to predict $\hat{c}_{i,j,t'}$ ($t' > 0$). We pre-process the arcs to retain arcs with a minimal flow and met other confidential requirements. We consider 71 origin FCs and 232 destinations. For each D_j, let $sorted_{j,t'} = (FC_{i',j}, FC_{i'+1,j}, \ldots)$, where $\hat{c}_{i',j,t'} \leq \hat{c}_{i'+1,j,t'} \leq \ldots \leq \hat{c}_{max,j,t'}$. The position of FC_i in $sorted_{j,t'}$ denotes the rank relative to D_j, $rank_{i,j,t'} = index(sorted_{j,t'} == FC_i) \in [0, 1, 2, ..., 70]$.

We see in Table 2 the rankings for D_0 alongside distance for two weeks, week 45 and 52. Of note, the ranks do not obey a distance law. In Table 3,

Rank	Week 45 FCs	Distance	Week 52 FCs	Distance
1	1	14.55	1	14.55
2	7	115.04	52	48.01
3	52	48.01	7	115.04
4	2	323.77	35	119.21
5	14	110.57	21	83.53
6	35	119.21	0	89.32
7	0	89.32	38	1378.55
8	21	83.53	31	137.60
9	31	137.60	2	323.77
10	38	1378.55	14	110.57

Table 2: Ranks for D_0. Rank 1 corresponds to the arc between an FC and D_0 with the lowest expected cost per package.

Rank	Week 45 FCs	Trucks	Week 52 FCs	Trucks
1	1	63	20	27
2	14	60	11	26
3	7	53	1	25
4	35	51	31	22
5	31	47	14	20
6	21	46	21	19
7	0	27	12	18
8	52	24	56	17
9	12	22	0	17
10	11	19	52	15

Table 3: Actuals for D_0. Rank 1 corresponds to the arc between an FC and D_0 with the largest number of trucks (we show only top 10). We only show actuals for the FCs considered in this study.

we show a glance of the top 10 week 52 actuals (see Table 4 for the number of connections per week). We see that the top 10 estimated ranks are similar to the top 10 actuals, except FC_{56}. New connections open as the peak season gets closer, so we expect peak-specific arcs to appear. Since the estimated ranks are time-dependent, then we expect some increase in ranks of infrequent arcs, but if a destination has a consistent set of 10 or more then we may not observe it in a top 10 rule. To take advantage of the recurring cycle of decisions and improve the chance we observe infrequent but necessary arcs, we propose a probabilistic model that updates based on user decisions using Thompson sampling as discussed next.

Week	Number of Connections
45	17
46	22
47	18
48	18
49	19
50	16
51	22
52	18

Table 4: Number of actual connections between the study FCs and D_0. Mean=18.75, Standard Deviation=2.19.

3.2 Thompson Sampling to Probabilistically Learn Future Arcs

The cascading bandit model for this problem is defined by $(N_D, N_{FC}, K_j, \theta)$, where N_D is the number of destinations, N_{FC} is the number of FCs, $K_j \leq N_{FC}$ is the number of arcs recommended at time t for D_j, and $\theta \in [0,1]^{N_{FC}}$ is a vector of connection probabilities. In a cascading scheme at time t, an agent recommends an ordered list $a_t \in \{1, \ldots, N_{FC}\}^{K_j}$. The user selects $k \leq N_{FC}$, call these selected arcs a'_t.

For arc $FC_i \rightarrow D_j$ denoted by $arc_{i,j}$, let $\alpha_{i,j} \equiv$ the number of times the arc was selected and $\beta_{i,j} \equiv$ the number of times the arc was not chosen, but was observed in the top K. Finally, to use the rankings as a weight for future decisions, let $rankpct_{i,j} = 1 - \frac{rank_{i,j}+1}{N_{FC}}$. In Algorithm 1, we describe the algorithm for using prior information to update the probability of selecting an arc, $arc_{i,j}$ at time t. Note that the initial probabilities of connections are sampled independently (a false assumption for network connections). However, we estimated ranks based on costs that have accounted for dependencies at some high level. Assuming overall structure of a network does not change significantly within short time intervals, then the geographic signatures presents a useful structured ranking. The updated connection probability reflects that information if it is reinforced by the user enough times. We show that this algorithm captures structure in a timely manner.

Algorithm 1 $ArcsTS(N_{FC}, K, \alpha, \beta)$

---

for $t = 1, 2, \ldots$ **do**
  \# Sample probability of connection;
  for $j = 1, 2, \ldots, N_D$ **do**
    for $i = 1, 2, \ldots, N_{FC}$ **do**
      Sample: $\hat{\theta}_{i,j} \sim Beta(\alpha_{i,j}, \beta_{i,j})$.
      Adjust $\tilde{\theta}_{i,j} = \hat{\theta}_{i,j} rankpct_{i,j}$.
    end
  end
  $a_t \leftarrow$ arcs with K largest $\tilde{\theta}_{i,j}$.
end
\# Update posterior distribution hyperparameters;
for $t = 1, 2, \ldots$ **do**
  \# Sample probability of connection;
  for $j = 1, 2, \ldots, N_D$ **do**
    for $i = 1, 2, \ldots, N_{FC}$ **do**
      $\alpha_{i,j} \leftarrow \alpha_{i,j} + \mathbf{1}\{arc_{i,j} \in a'_t\}$.
      $\beta_{i,j} \leftarrow \beta_{i,j} + \mathbf{1}\{arc_{i,j} \in a_t\}$.
   **end**
  end
end

3.3 Experimental Results

Using 8 weeks of data, we run Algorithm 1 to recommend 10 arcs. The probabilities can be initialized using $\alpha_{i,j} = 0.1$ and $\beta_{i,j} = 1.0$. Choosing an arc increases α. If an arc is not chosen but in the top K rule, then β increases. For the initial week (week 45), the sampled probabilities of connection are uninformative, so on a first initialization we can use the ranks. In Figure 6, by the fourth week, FCs with rank percentiles greater 50% start to get recommended with probability greater 50%. FCs with rank percentiles below 50% tended to be neither selected nor viewed, so by the 5th week had low probabilities (no rewards).

Table 5 shows the 10 arcs with largest sampled probability of connection to D_0. F_{56} is recommended among the top 10. Although this is by probability, this case shows the usefulness of Bayesian reinforcement based on estimated ranking and operator decisions. FC_{69} was not in the top 10 actuals for D_0, but it was active with only 2 trucks.

One way to decrease the role of chance in the top K rule (here $K = 10$) is to introduce a top K_j rule, where the number recommended depends on the jth destination. D_0 alone averaged about 19 connections (sd=2.19) over the 8
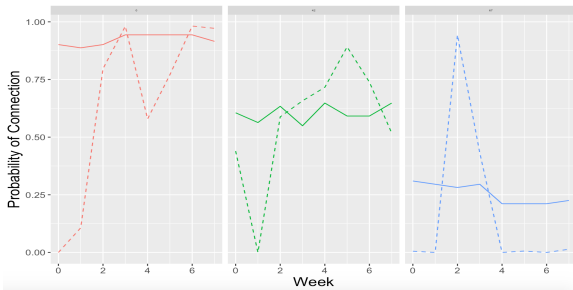
Fig. 6: Rank percentile (solid) and connection probabilities (dashed) with D_0 across 8 weeks for 3 random FCs (0, 42, 67).

Rank	Week 52 FCs	Distance
1	52	48.01
2	1	14.55
3	7	115.04
4	0	89.32
5	35	119.21
6	2	323.77
7	22	133.57
8	56	340.90
9	69	1487.01
10	14	110.57

Table 5: Ranks for D_0. Rank 1 corresponds to the arc between an FC and D_0 with the largest sampled probability of connection.

weeks. Other destinations play smaller or larger roles in terms of arc connectivity depending on time and network position. One approach to choosing K_j is to use the probabilities of connection themselves. We can treat arc connectivity probabilities as belonging to a Bernoulli random variable, $Y_{i,j} \sim Bernoulli(\theta_{i,j})$. Let $Y_j = \sum_i Y_{i,j}$. Assuming independence among $Y_{0,j}, \ldots, Y_{70,j}$, the expected number of arcs, N_j^{arc}, can be calculated as $N_j^{arc} = \mathbb{E}(Y_j) = \sum_i \theta_{i,j}$. In this way, we can use $\tilde{\theta}_{i,j}$ to estimate $\theta_{i,j}$ and say the agent wants to recommend $K_j = N_j^{arc}$ arcs. Under this formulation, we can also estimate the variability of the agent's recommendation total on destination j as $\mathbf{Var}(Y_j) = \sum_i \theta_{i,j}(1 - \theta_{i,j})$. For D_0 over the 8 weeks, the average N_j^{arc} was 16 (sd=$\sqrt{\mathbf{Var}(Y_j)}$ = 2.31). During week 52, the agent would have recommended 24 arcs compared to the actual 18.

Overall, this shows the agent is already converging on network behavior for D_0 as far as recommendations might be needed to. As a recommendation agent, we primarily want to reduce the search space for the next set of connections. We also want to be able to use the ranks and probabilities of connection to experiment with new networks. As discussed in Section 2.4, the geographic signatures can be used to summarize the fundamental properties of different networks. As shown here, these geographic signatures can be used to rank arc connections within a network. These ranks in turn can influence a cascading model's probability of connection to recommend a set of arcs. A simple framework for user-driven agent-assisted testing for the potential of a new network addition can be formulated as follows:

1. Fit a model that captures important network dependencies using historical arc costs/flows data.
2. Predict cost/flow on artificial data for new connections.
3. Use predictions to rank old and new connections.
4. Initialize new arcs using most similar existing arcs.
5. Run the agent and see how close new arcs are to be included in the set K_j. With a range of artificial data sets, uncertainty of new arc utility can be measured.

There is more that can be done within Algorithm 1. For instance, a discount factor can be added so that connections that are guaranteed converge at some high probabiltity. Moreover, it would be important to also introduce penalties for inactivity. One simple idea is to create a rule dependent on decreasing rank: lower β by 1 if an arc drops in rank by some threshold. Finally, the most important addition to such an algorithm would be to capture the dependencies amongst the probabilities of connection in a way that still allows user-driven experiments.

4 Discussion

In this paper, we discussed image transformations for learning recurring decisions. We introduced summarizing complex geographic network structures using geographic summaries or geosigs. Geosigs captured fundamental network properties that can be helpful in ranking the potential of new connections. For recurring connectios, we introduced a Bayesian reinforcement framework where an agent probabilistically determines connections based on ranks reflecting network value and user

decisions. There is more that can be considered in future work both in terms of method and application. The probabilities and rankings can be merged into one probabilistic model that captures network dependencies more explicitly. Furthermore, tendencies in the agent's decisions can be used to track emerging networks in applications beyond transportation.

5 Declarations

On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

[1] Osama K Al-Shaykh and Russell M Mersereau. Lossy compression of noisy images. *IEEE Transactions on Image Processing*, 7(12):1641–1652, 1998.

[2] Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016.

[3] Guido Berning, Marcus Brandenburg, Korhan Gürsoy, Vipul Mehta, and Franz-Josef Tölle. An integrated system solution for supply chain optimization in the chemical process industry. *OR spectrum*, 24(4):371–401, 2002.

[4] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 891–900, 2015.

[5] Quanyu Dai, Qiang Li, Jian Tang, and Dan Wang. Adversarial network embedding. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[6] Christodoulos A Floudas and Chrysanthos E Gounaris. A review of recent advances in global optimization. *Journal of Global Optimization*, 45(1):3–38, 2009.

[7] Martin Grunow, Hans-Otto Günther, and Matthias Lehmann. Campaign planning for multi-stage batch processes in the chemical industry. *Or Spectrum*, 24(3):281–314, 2002.

[8] Josef Kallrath. The concept of contiguity in models based on time-indexed formulations. *Scientific computing in chemical engineering ii*, pages 330–337, 1999.

[9] Josef Kallrath. Solving planning and design problems in the process industry using mixed integer and global optimization. *Annals of Operations Research*, 140(1):339–373, 2005.

[10] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, and Le Song. Learning combinatorial optimization algorithms over graphs. In *Advances in Neural Information Processing Systems*, pages 6348–6358, 2017.

[11] Carlos A Méndez and Jaime Cerdá. An milp-based approach to the short-term scheduling of make-and-pack continuous production plants. *Or Spectrum*, 24(4):403–429, 2002.

[12] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710, 2014.

[13] Daniel J Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, Zheng Wen, et al. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11(1):1–96, 2018.

[14] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th international conference on world wide web*, pages 1067–1077, 2015.

[15] Francis J Vasko, JD Bobeck, MA Governale, DJ Rieksts, and JD Keffer. A statistical analysis of parameter values for the rank-based ant colony optimization algorithm for the traveling salesperson problem. *Journal of the operational Research Society*, 62(6):1169–1176, 2011.