Graph Transformer for Workforce Planning

Jeremy Curuksu
People Experience and Technology
curukj@amazon.com

Jimmy Rigby
People Experience and Technology
rigbjame@amazon.com

Abstract

We present a Graph Transformer deep learning method for workforce planning which can identify potential talent risks and forecast individual monthly talent movements in each segment of the Amazon corporate population. This method outperforms linear methods in current usage at Amazon by 40% in 76% of the segments, and outperforms all state-of-the-art baselines tested by 15% in 61-70% of the segments. Given the dependency Graph in the proposed method can be used to interpret and understand talent forecasts, there is little drawback to using this method compared to using linear trailing rates for workforce planning.

1 Introduction

Workforce planning at Amazon sets year-end headcount targets by financial cost center to meet the company's current and future staffing needs based on business goals and talent movement forecasts (hires, promotions, transfers, attritions). Individual team leaders further plan their workforce needs by individual team, job type, job level and location. Failure to accurately forecast future headcounts and staffing needs often result in delays in productivity and costly resource allocation [1].

Forecasting Amazon talent movements is challenging due to complex spatial and temporal dependencies within the Amazon population, and non-stationarity that result from unusual events such as the Covid pandemic [2]. Amazon is an especially difficult forecasting problem due to its diverse operation workforce and unprecedented size (over 1.6M employees in peak season [2]).

Examples of talent movement dependency at Amazon include talents in similar environments, with similar profiles and job market opportunities, or under similar talent management strategies. Many other factors, including external factors such as large swings in the company's stock value [3], can influence talent flows and thereby create correlated traffic patterns in the Amazon population.

Forecasting spatial and time-dependent data in large, complex traffic networks was recently addressed by estimating a dependency Graph that parsimoniously represents the spatial dependency between different locations in the network (nodes), and using this Graph to sparsify a deep learning Transformer model [4]. Transformers [5, 6] have been shown to significantly outperform RNN and CNN and improve the learning of long-range temporal dependency [5]. By assigning each neuron of the Transformer with a spatial location and using knowledge from the dependency Graph to prune neural connections that are not dependent, the resulting *Graph Transformer* could efficiently capture *both* spatial and temporal dependencies and scale to large traffic networks [4].

In this paper, we use a multivariate Gaussian approximation to find the dependency Graph of talent movements over the different teams of the Amazon corporate population defined by leader, job type, and job level. We use this Graph to derive insights into the overall dynamics of Amazon talent movements and identify potential drivers of talent risks. We then integrate the Graph in a Transformer model, and forecast individual talent movements for each team (node) of the Amazon corporate population. We demonstrate the forecasting performance improvement compared to baseline methods in current usage at Amazon (linear trailing rates) and more advanced baseline forecasting methods. Results show that our method outperforms linear methods by 40% in 76% of the nodes, and all state-of-the-art baselines tested by 15% in 61-70% of the nodes.

2 Methodology: Graph and Transformer for workforce planning

We describe the multivariate Gaussian approximation method used to determine the dependency Graph among individual teams within the Amazon corporate population (section 2.1). Then we describe the application of Transformer models for multivariate time series forecasting (section 2.2). Finally, we combine the two together into an original Graph Transformer model to forecast individual talent movements for each team of the Amazon corporate population (section 2.3).

2.1 Analyzing talent movements with Graph

We use a multivariate Gaussian approximation to determine the dependency Graph of 5 talent movements (hires, promotions, transfers, regretted attritions, unregretted attritions) over the 72 teams existing in the Amazon corporate population when defined by leader (VP level), job type and job level. A Gaussian approximation models the spatial and time-dependent data as the following multivariate distribution over n nodes (5 talent movements $\times$ 72 nodes $= 360$ nodes $= n$):

$$f(x_t) = (2\pi)^{-\frac{n}{2}} \Sigma^{-1} \exp\left(-\frac{1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu)\right)$$

where μ is the mean vector and Σ^{-1} the inverse of the covariance matrix, called the precision matrix. In this paper, we estimate the precision matrix directly from the data by Graphical Lasso [7, 8]. The precision matrix characterizes the conditional dependency between different nodes. Two nodes may be dependent if their talent movements are correlated. But correlation does not guarantee dependence. A correlation may occur because the two nodes are jointly dependent on a third node (Fig 1). In contrast, conditional dependency measures the correlation between two nodes after controlling for the talent movements in other nodes (i.e., keeping them constant). We measure the conditional dependency between nodes i and j by their conditional correlation coefficient:

$$C\left(x_t^i, x_t^j | x_t^{-ij}\right) = -\frac{\Sigma_{ij}^{-1}}{\sqrt{\Sigma_{ii}^{-1}\Sigma_{jj}^{-1}}}$$

Following [4], we preset a threshold on each entry of the conditional correlation matrix and treat nodes i and j as conditionally dependent if the absolute value of $C\left(x_t^i, x_t^j | x_t^{-ij}\right)$ is above the preset threshold. The non-zero entries define the structure of the dependency Graph between nodes, which we then use to sparsify a Transformer model (section 2.3).

2.2 Forecasting time-dependent talent movements with Transformer

To forecast talent movements, we implemented a Transformer encoder-decoder architecture [5, 6] which has been shown to outperform state-of-the-art baselines in sequential tasks for processing natural languages [5] and forecasting time series [4]. Transformer relies on *Attention* functions to encode and learn the relationship between distant positions in a sequence. It does not use recurrence nor convolution. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors [5]. The output of an attention function at position i in a sequence is defined by the following scaled dot product, where a query vector Q_i attends to all other positions j as a weighted sum over all these positions:

$$\text{Attention}_i = g\left(\frac{Q_i K^T}{\sqrt{d}}\right) V$$

Where V is the matrix of value vectors V_j and d the dimension of the query and key vectors. A value vector can have arbitrary embedding dimension, which also defines the dimension of the output. In multi-layer encoder-decoder architectures, this output is used as the value vector at the next layer.

The *encoder* consists in *self-attention* layers i.e., it relates different positions of a given sequence in order to compute a representation of the sequence. Each query Q_i at position i can attend to each other position j in the sequence as much or as little as the learning task requires by weighting each position with a key K_j. The positions themselves can provide any type of information by being represented as a value vector V_j. As a result, an attention mechanism can learn *long-range* temporal dependencies just as easily as shorter range ones, simultaneously, in contrast to RNN and CNN.

For n input time series of m steps, the encoder embeds the sequences of historical values into a context tensor of dimension (l, m) where l is the embedding dimension. The number of input time series n is not limited to the numbers of Amazon teams for which we need forecasts: it can also include auxiliary information. In this paper, we include a measure of *S&P 500* stock prices expected future volatility to reflect external market conditions known to impact talent movements [3].

The *decoder* consists in self-attention layers which allow each position in the decoder sequence (the forecasted horizon) to attend to all positions in this sequence up to and including that position, and also an *encoder-decoder attention* layer*,* in which the queries come from the previous decoder layer, and the keys and values come from the output of the encoder [5]. The final output of the decoder is a forecast for the next timestep based on past sequences (encoder sequence) and on all previous time steps in the target horizon (decoder sequence). More details can be found in [5,6] for the original Transformer architecture and in [4] for the one used in this paper to forecast time series data.

The key limitation of an attention function is its reduced resolution due to averaging attention-weighted positions; in complex sequences, different attention maps can produce an identical output [5]. The original Transformer architecture addressed this problem by using multi-head attention. Multi-head attention repeats the scaled dot product described above and average the results together, which allows the model to jointly attend to information from different representation subspaces at different positions. Instead of averaging over multiple heads, it would be even more beneficial if we knew how to focus the attention heads on connections that matters the most. This is precisely what the dependency Graph presented in section 2.1 can help with.

2.3 Combining Graph and Transformer to forecast large spatial and time-dependent data

We use the dependency Graph of Amazon talent movements presented in section 2.1 to sparsify the architecture of the Transformer presented in section 2.2. This focuses the parameterization (i.e., learning) of attention heads to connections for which entries in the conditional correlation matrix are above a preset threshold. This sparsification of the neural network enables Transformer to better capture the dependencies between Amazon teams (nodes) in term of talent movements.

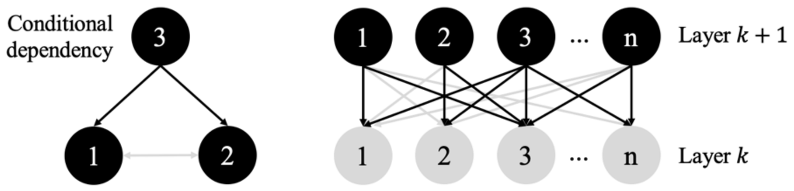

Figure 1: Example of co-dependent nodes (left) and sparsified neuronal connections (right)

To integrate the Graph into the encoder-decoder architecture, we associate each neuron in the Transformer with a node in the Graph (Fig 1) or auxiliary information, then prune connections between neurons associated with nodes not connected in the Graph. This helps better capture dependencies between talent teams because a team in this learning system is impacted only by itself and other clearly dependent teams. In addition, this Graph-based sparsification regularizes the overall Transformer architecture, which helps minimize general overfitting problems.

3 Graph analysis of Amazon talent movements

In this section, we present insights derived from Graph analysis using multivariate Gaussian approximation on the dynamics of talent movements between the 72 teams existing in the Amazon corporate population when defined by leader (VP level), job type and job level. In this paper, we make no mention of actual leader names and use symbols instead ($x_1, x_2 \ldots$). We describe how our method can be used to address three talent questions: Which teams in the Amazon corporate population are related in term of talent movement (3.1), which teams are the most interdependent (3.2), and for a given talent movement in a given team, which other teams are most related (3.3).

3.1 Which teams in the Amazon corporate population are related in talent movement?

In this section and the next, we use transfers, one of the five general talent movements, as illustrative example. Similar conclusions could be derived from the Graph for the other four talent movements, and section 3.3 will describe how to derive insights into the relationship *between* talent movements. Fig 2 (left) shows the complete transfer conditional correlation matrix organized by job level.

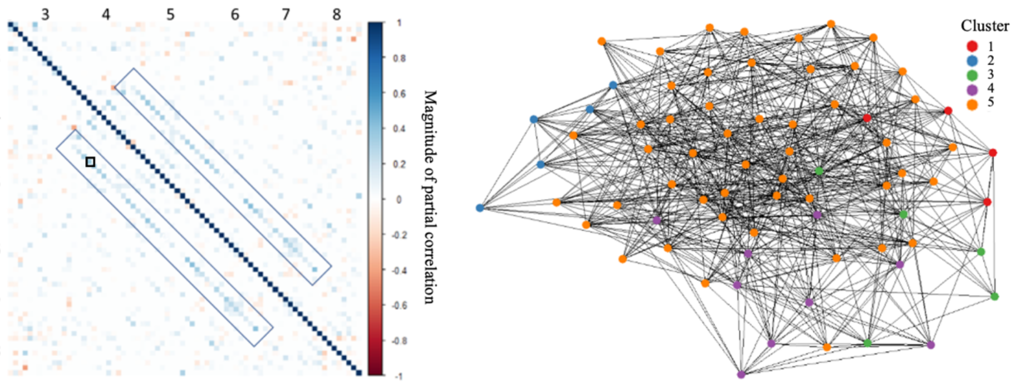

Figure 2: The Amazon corporate talent transfer dependency Graph

The blue entries in the transfer matrix that parallel the main diagonal suggest that segments who are separated by one job level but fall under the same leader and job type are related in internal transfer. For example, the black box shows the correlation between level 4 and level 5 software development engineers (SDE) reporting to leader x_1: its dark blue color means that when a level 5 SDE team reporting to leader x_1 had higher transfers, higher transfers were also observed in level 4 SDE teams reporting to leader x_1. This correlation is not observed when the difference in job level is two or more, such as level 3 or 6. In practice, this type of insight could inform transfer policies by better anticipating the business impact of high (or low) transfer rates across specific segments of the Amazon population.

3.2 Which teams are the most interdependent in the Amazon corporate population?

Visualizing the overall talent dependency Graph can reveal "clusters" in the Amazon population (Fig 2, right) where talent movements are more dependent within the cluster than with movements in other clusters i.e., relatively independent from the rest of the organization. For example, the cluster #4 tagged with purple nodes in Fig 2 corresponds to talents of level 6 and level 7 for whom internal transfers appear more dependent on one another than on the rest of the organization.

Table 1 reports the top five most- and least-dependent teams in the transfer dependency Graph. SDE and Tech talents with level 4-5 and reporting to leader x_1-x_2 represent a typical cluster of employees i.e., transfers in this cluster are most likely to impact transfers in other teams of this cluster.

Table 1: Transfer nodes most-/least-related in Amazon corporate population

Five most related nodes	Weight	Five least related nodes	Weight
SDE, level 4, leader x_1	2.40	Man, level 7, leader x_2	2.24
Tech, level 5, leader x_1	2.36	Tech, level 3, leader x_2	0.84
SDE, level 4, leader x_2	2.32	Tech, level 3, leader x_1	0.73
Tech, level 4, leader x_2	2.30	SDE, level 8, leader x_3	0.62
Tech, level 6, leader x_1	2.30	Man, level 8, leader x_1	0.46

In practice, identifying clusters of talent movements help develop talent strategies valid *across* multiple clusters e.g., promotions and hires targets, as well as strategies tailored to specific clusters.

3.3 For a given talent movement in a given team, which other teams are most related?

Anticipating the magnitude and direction of talent movements following a talent event observed in a given team can also help organization leaders design and implement focused talent strategies. For example, Table 2 identifies the five most informative nodes related to SDE level 6 in leader x_1 organization in term of internal transfers. For leader x_1, SDE level 5 talent transfers are the best indicator of SDE level 6 transfers ($r = 0.192$). Interestingly, SDE level 6 under leader x_2 are also significant indicators but are negatively correlated ($r = -0.107$). The negative correlation might be because SDE level 6 talents tend to transfer in and out of $x_1$and x_2 organizations.

Table 2: Transfer nodes most related to SDE L6 x_1 node

Five most related nodes	Weight
SDE, level 5, leader x_1	0.192
HVH, level 3, leader x_2	0.187
Tech, level 5, leader x_1	0.134
Tech, level 5, leader x_2	- 0.122
SDE, level 6, leader x_2	- 0.107

To further investigate this negative relationship, Fig 3 shows the number of transfers occurring in $x_1$and x_2 organizations with the positive (green) and negative (red) correlations relative to SDE level 6 under leader x_1 (black). When a significant peak or trough in transfers happens for SDE level 6 under leader x_1, we observe that a similar peak or trough generally happens too in the two other teams identified, in opposite direction as suggested by the sign of the correlation coefficient.

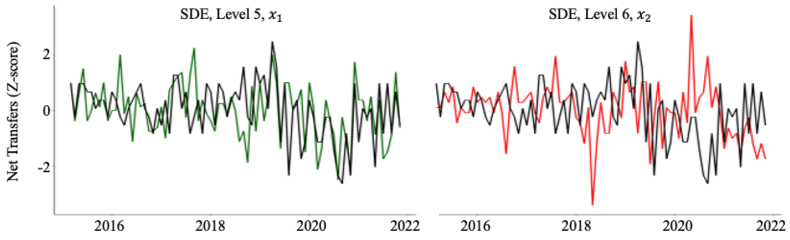

Figure 3: Net transfer in SDE level 6 x_1, level 5 x_1, and level 6 x_2 nodes

In practice, further investigation of the time lags of these peaks and troughs could be carried out to identify potential systematic precursors of specific transfers. More generally, anticipating the magnitude and direction of talent movements related to a talent event observed in a given team helps talent planners coordinate specific talent movements in the teams that they support.

Finally, the dependency Graph was also estimated across all five talent movements for the entire Amazon corporate population. Fig 4 summarizes the hire-termination section of the Graph by aggregating across all job types for leader x_1.

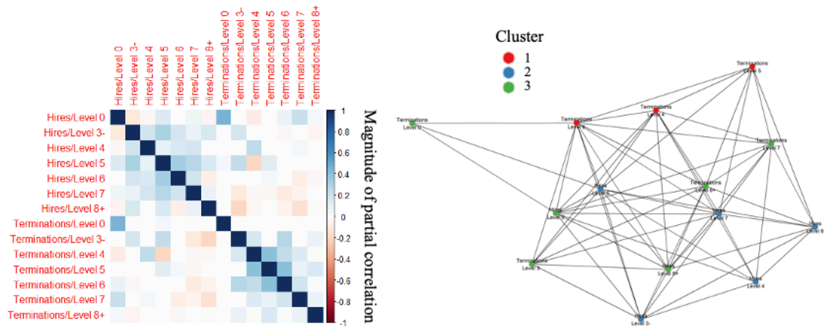

Figure 4: Hire-termination dependency Graph aggregated across job types for leader x_1

Fig 4 shows one of the greatest dependency across hires and terminations in the Amazon corporate talent population under leader x_1 is between level 7 hires and level 6 terminations. This correlation is negative. A possible interpretation of this observation is that directors (level 7) who just got hired have a significant impact on retention of their direct reports, most of whom are likely leaders in a level 6 position, by implementing a new, clear and innovative executive vision. In situations where these director positions remain unfilled, level 6 managers might lack clear directions coming from a direct, newly dedicated director. This interpretation is given just for illustration purposes. In practice, identifying strong dependencies as show in Fig 2-4 and Tables 1-2 represent only a starting point to help focus experimentation and further investigation into potential causal effects.

4 Forecasting Amazon talent movements

Finally, we present results from Graph Transformer to forecast individual talent movements in each team of the Amazon corporate population. We describe the cross validation set up and baseline models used for benchmarking (section 4.1), then summarize the benchmarking results (section 4.2).

4.1 Cross-validation and benchmark set up

The Graph Transformer performance was evaluated on the task of forecasting Amazon talent movement 12-month horizons for a given node by auto-regressing on the most recent 18 months (input sequences) across all 72 nodes, using a 1-month expanding window training and cross-validation approach [10]. A monthly time series ranging from 01/2015 to 04/2022 was assembled for the number of promotions, regretted/unregretted attritions, transfers and hires for the 72 teams of the Amazon corporate population defined by leader, job type and job level. The concatenated auxiliary data was the S&P 500 stock prices expected future volatility index (VIX). This represents a total of 5 talent movements $\times$ 87 months $\times$ $(72 + 1) = 31{,}755$ data points. We split this data set into training vs. test set and tuned hyperparameter exclusively in the training set to maintain the integrity of the test set. Hyperparameters tuned included the length of the input sequence (up to 18 months), the number of hidden units and attention heads, and the Graph threshold for connection pruning. The training set used data from 01/2015 to 04/2020 which represents over 40 pairs of input-output sequences per talent movement per node. The test set used data from 12/2019 to 05/2022 which represents over 17 pairs of input-output sequences per talent movement per node i.e., well over 6,000 output test data points across all nodes. All output sequences are defined in 2021-2022 so there is no overlap with output sequences used for training the models.

To benchmark the performance of Graph Transformer at forecasting Amazon talent movements, we used a trailing 3-month extrapolation method as baseline because this is still currently used by many teams at Amazon for workforce planning. In this model, an input 12-month sequence is used to derive an average 3-month rate which is in turn used to forecast the next 12 months.

We also evaluated more advanced forecasting methods based on s-ARIMA, state space and Prophet models. s-ARIMA (Seasonal Autoregressive Integrated Moving Average) predicts future values of a time series by regressing on preset lags and seasonal periods of both the series itself and of random fluctuations around the series moving average [11]. The lags and seasonal periods are determined by hyperparameter tuning. A similar state space model with additional parameters tuned to allow for multiplicative effects of seasonality and random fluctuations was also evaluated. Finally, Prophet [12] is an additive model that uses a linear piecewise function to decompose each series into trends and change points, adds weekly seasonality using a Fourier Transform, and also adds custom special events such as holidays. See [12] for more details.

The performance of each model was measured using the average Mean Absolute Percent Error (MAPE) calculated over all 12-month output sequences in the test set for each talent movement and each of the 72 nodes, as described above. We excluded promotion forecasts in the results reported below because all models tested have disproportionality worst performance for promotion forecasts compared to other talent movements, which significantly biases MAPE averages. Promotions are completely under the control of leaders and managers at Amazon, which is not the case of other talent movements. Further development to optimally forecast promotions is needed and underway.

4.2 Forecasting performance results

Table 3 shows the average MAPE calculated on the test set for the Graph Transformer model and each of the benchmarked models described in section 4.1. To further characterize the performance obtained, we also report the proportion of nodes for which an improvement in MAPE is observed relative to each baseline, and the observed MAPE specifically in these nodes.

Table 3: Performance of Graph Transformer to forecast talent movements

Model	MAPE	Nodes improved	MAPE in nodes improved
Graph Transformer	84%	N/A (*self*)	93%
Prophet	95%	70%	108%
s-ARIMA/State Space	87%	61%	107%
Trailing 3-month	112%	76%	133%

All advanced methods perform better across all metrics compared to the trailing 3-month baseline. This is not surprising given the latter cannot model well the non-linearity and non-stationarity within the data, which were very pronounced during the Covid pandemic period, and also does not consider the spatial dependencies between locations in the structure of its coefficient matrices.

The biggest improvement is observed for the Graph Transformer with a MAPE of 84%, representing a decrease in MAPE of 3 and 11 percentage points relative to s-ARIMA/State Space and Prophet models, respectively. The proportion of nodes improved is 61% and 70% relative these methods, with a decrease in MAPE of 15% in these nodes, which confirms that Graph Transformer outperforms other methods for the general talent movement forecasting task and not just for a small number of specific nodes. Compared to the simplest trailing 3-month baseline, an improvement in 76% of nodes is observed. This corresponds to a decrease in MAPE of 28% across all nodes, and a decrease in MAPE of 40% across the improved nodes.

Different levels of performance were observed depending on which talent movement was forecasted (e.g., promotion, see section 4.1). Further optimization is currently underway to identify hyperparameters optimal for specific types of nodes and talent movements.

5 Conclusion

We developed a Graph Transformer deep learning architecture for workforce planning at Amazon, and used it to (1) identify related talent movements across the different teams of the Amazon corporate population defined by leader, job type and job level, and (2) to forecast individual talent movements 12 months in the future for each team.

By identifying related teams in the Amazon talent population, the Graph can be used in workforce planning to better anticipate the scope of observed talent events, identify potential risk factors, and help focus experimentation and further investigation into potential causal effects.

Individual talent movement forecasts are needed in workforce planning for Amazon to meet current and future staffing needs. Our results demonstrate that the method presented in this paper outperforms linear methods typically used in workforce planning at Amazon (trailing rates) by 40% in 76% of the corporate population segments, and outperforms all state-of-the-art baselines tested (s-ARIMA/State Space, Prophet) by 15% in 61-70% of corporate population segments.

Future research is needed to identify hyperparameters optimal for specific types of nodes and talent movements, to include explicit locations, and to extend Graph Transformer to other Amazon populations such as the more versatile and fast-paced field population in fulfillment centers.

It is worth noting that even though the trailing 3-month baseline method cannot model well the non-linearity and non-stationarity of Amazon talent data, which explains the disproportionality high increase in performance when using any of the advanced time series forecasting methods, it is actually the method in current use for most workforce planning decisions at Amazon.

Given that the dependency Graph in our method can be used to interpret and understand talent movement forecasts, there is little drawback to using the method presented in this paper compared to using linear trailing rates for workforce planning. In the future, we will add an AI interpretability process [9] to the Graph Transformer model to explain each forecast, individually. We hope that the results presented in this paper will contribute to a wider adoption of machine learning methods for workforce planning at Amazon.

Customer Problem Statement

Amazon team leaders, talent planners and financial analysts forecast future e.g., end-of-year, talent movements so they can meet future staffing needs. Failure to accurately forecast future headcounts and staffing needs result in delays in productivity and costly resource allocation. Talent planners and team leaders compare talent movement forecasts with plans defined by financial analysts to assess the year-end talent *gaps-to-goal*. They also create risk scenarios to measure how changes in specific talent drivers impact forecasted headcounts, to identify talent movements at risk of driving gaps-to-goal, and to plan for recruiters needed to support hiring. The method presented in this paper identifies Amazon talent movements that are most related to help identify potential risk factors and design strategies to close gaps-to-goal. It also forecasts future talent movements in individual segments of the Amazon population. Forecasting talent movements is challenging at Amazon due to complex spatial and temporal dependencies, and due to non-stationarity which was most recently induced by the Covid pandemic.

References

[1] Safarishahrbijari A (2018) Workforce forecasting models: a systematic review. Journal of Forecasting 37: 739-753

[2] Pisal S. (2021) Rise of Facebook, Amazon, Apple, Netflix, Google during COVID-19 pandemic. MSc diss., California State university, 1311

[3] Xu H, Yu Z et al. (2018) Dynamic talent flow analysis with deep sequence prediction modeling. IEEE Transactions on Knowledge and Data Engineering 31: 1926-1939

[4] Yang L, Moura J (2019) Forecaster: A Graph transformer for forecasting spatial and time-dependent data. arXiv preprint arXiv:1909.04019

[5] Vaswani A, Shazeer N et al. (2017) Attention is all you need. Advances in neural information processing systems, 30: 5998-6008

[6] Dai T, Yang Z et al. (2019) Transformer-xl: Attentive language models beyond a fixed-length context. arXiv preprint arXiv:1901.02860

[7] Friedman J, Hastie T, Tibshirani R (2008) Sparse inverse covariance estimation with the graphical lasso. Biostatistics, 9: 432-441

[8] Städler N, Bühlmann P, Van De Geer S (2010) L1-penalization for mixture regression models. Springer Test 19: 209-256

[9] Frye C, Rowat C, Feige I (2020) Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability. Advances in Neural Information Processing Systems, 33: 1229-1239

[10] Cerqueira V, Torgo L, Mozetič I (2020) Evaluating time series forecasting models: An empirical study on performance estimation methods. Machine Learning, 109: 1997-2028

[11] Kaushik I, Singh SM (2008) Seasonal ARIMA model for forecasting of monthly rainfall and temperature. J Environ Res Dev 3: 506-514

[12] Taylor SJ, Letham B (2017) Forecasting at scale. Journal of American Statistical Association, 72: 37-45