PFI-TT: Democratizing Recommender Systems for E-commerce with Automated Graph Neural Networks.

Proposal submission on FastLane.nsf.gov [DEADLINE: February 10, 2021.]

- ☐ Project Summary [One (1) page max].
- ☐ FastLane documentation [Collaborators and other affiliations, Bio sketch, Budget, Data management, Equipment and facilities, etc.]

Project Description. -Fifteen (15) pages max-

- ☐ Executive Summary (no more than one page)
- ☐ From NSF Basic Research to Addressing a Market Opportunity (suggested length: 4 -5 pages)
- ☐ Technical Challenges and Applied Research Plan (suggested length: 5-7 pages)
- ☐ Achieving Societal Impact through the Realization of Commercial Potential (2-4 pages)
- ☐ Project Team (suggested length: 1-2 pages)
- ☐ Partnerships (suggested length: 1-2 pages)
- ☐ Training Future Leaders in Innovation and Entrepreneurship (suggested length: 1-2 pages)
- ☐ Broadening Participation (suggested length: up to 1 page)

Others

- ☐ Letters of support (max. 3)

Project summary -1 page Alfredo 'Done'

1 Overview

This project aims to build upon our NSF award (#1750074), CAREER: Human-Centric Big Network Embedding grant, to further translate our academic research and technologies in Automated Graph Neural Networks into advanced recommendation systems to help small businesses adopt advanced machine learning (ML) technologies. The e-commerce industry is already heavy users of enterprise business analytics software. Online stores are exceptionally ready to adopt enterprise software because they already heavily use information technology as a core part of their business. Many large enterprises can currently implement recommendation systems; however, existing approaches focus on heuristics to decide the optimal configuration of complex pipelines. This project investigates a novel direction to explore Graph Neural Networks (GNN) to build complex recommendation systems. The proposed project's primary goal is to use our work in Automated GNN (AutoGNN) to develop advanced recommendation systems tailored to be designed and deployed by programmers with basic ML knowledge. This translational research's successful outcome will also lead to ML interpretation advancements for users to learn the mechanism behind the recommendation system selections and why the system made specific recommendations. This reduction in complexity and deployment cost can help subject matter experts to understand, trust and leverage advanced ML technologies. Simultaneously, the work proposed here can accelerate the technology adoption for a larger group of businesses that can now embrace the data revolution to start using their data assets to innovate e-commerce in new ways.

2 Intellectual Merit

This project aims to enhance the efficiency of developing an ML algorithm, targeting some practical industrial challenges. Our proposed AutoGNN framework requires significantly less computational resources and human effort, thereby serving as an alternative approach for small businesses. In particular, this project aims to tackle the following challenges:

- **Any Data Format to Graph Data (Any2Graph)**: Discover a graph-structure from real-world non-graph format data, such as tabular, images, and time-series data in the recommender system. The graph structure can represent a relationship between different instances or different features of each instance, which is essential for downstream tasks.
- **Graph Neural Architecture Aearch (GraphNAS)**: Search an optimal GNN architecture specific for the desired task in an automated way, reducing human experts' dependence. The proposed GraphNAS involves the meticulous design of search space, search strategies, and each GNN architecture evaluation.

Our team is looking to apply this project's techniques to different areas, such as recommender systems, time-series analysis, and feature interaction modeling in tabular data. Specifically, for recommender systems, the proposed GraphNAS generates the optimal GNN architecture to learn the item's representation, which is crucial for recommendation performance. For time-series analysis, our team is planning to utilize the proposed Any2Graph to model the relationship between multiple time-series, which plays a pivotal role in the performance of downstream tasks, such as time-series classification and clustering, and regression. For feature interaction modeling, our team proposes applying Any2Graph to quantify the interaction effect between any two features, which is crucial for improving downstream tasks' performance.

3 Broader impacts

This PFI-TT project will expand the fundamental understanding and provide practical computational tools in dealing with an emerging and critical ML automation and interpretation problem for recommendation systems with significant applications in e-commerce. The proposed translational research's successful outcome will enable more companies and researchers to build and deploy advanced recommendation system frameworks but at a fraction of current time and cost. The strong support from companies such as Amazon and Samsung has been instrumental in our search to commercialize our recent findings and ML technologies. Such industrial partnerships have made evident that successful human-technology partnerships could effectively enhance the US e-commerce experience to simplify excessive amounts of data and offer more relevant products to online shoppers.

Recommenders are essential online channels: they shape the media we consume and the products we seek. And yet, recommenders may be subject to algorithmic bias that can lead to negative consequences in recommendations offers. One clear example is that ad recommenders can exhibit racial discrimination. Overcoming such algorithmic bias is a top priority for our team and the reason why we have developed and implemented fairness-aware algorithms that we designed to maintain quality while dramatically improving fairness in recommendation systems. Our vision is to drive our research results into the market and serve as an example to help drive a more ethical ML use in e-commerce.

1 Executive Summary

1.1 The Customer

Small businesses and early-stage startups worldwide are currently looking to improve their online operations to serve a market shifting to consume mainly on the web, which is often an overlooked collateral effect of the current stay-at-home restrictions worldwide. Recommendation systems are a valuable tool in social media for startups because they reduce data overload by providing meaningful results to customers that increase engagement. We want to highlight that online-based startups are already a heavy user of enterprise software. Online businesses are particularly prepared to adopt enterprise software because they already heavily use information technology as a core part of their business.

1.2 The Value Proposition

Personalized recommendation systems are ubiquitous and the primary source of revenue for many online services such as e-commerce, advertising, and social media. At its core, a recommendation system estimates how likely a user will adopt an item based on historical interactions like purchases and clicks. While recommender systems have become an important computational component in major giant social media companies, their complicated nature prevents early-stage startups from easily adopting and understanding powerful ML algorithms and recommendation systems in their daily online operations.

The project proposed here integrates Automation and Interpretability into ML-based recommender systems. Our goal is to reduce the ML development speed and complexity to accelerate the adoption of advanced recommendation systems by early-stage startups and help them thrive in the digital-first economy. Our team has already developed a tool that allows our current users to obtain comparable recommendation systems like those designed by human experts in large enterprises but at a fraction of current time and cost.

1.3 The Innovation

This project builds upon *our highly praised open-source system, AutoKeras*, which has become one of the most used AutoML systems (with over 7,700 stars and 1,300 forks on Github), to provide an automatic and general graph representation learning framework for various real-world applications, such as recommender systems, time-series analysis, and interaction detection. Our proposed framework learns the graph representation for any modal data to reveal the intrinsic relatedness of data at the feature or instance level. Additionally, domain experts can automatically and efficiently develop the ML system with less human effort.

Our team introduces **a graph learning framework** (Any2Graph) to transform data of any modality to a graph structure, a powerful tool to model the data's intrinsic relatedness. For example, the features are separately listed in tabular/time-series data leading to implicit-relatedness representation. For any specific application task, Any2Graph can tailor the graph structure in an end-to-end manner, reducing the ML system complexity with comparable performance.

Graph neural architecture search (Graph NAS) is another innovation of this project. The current GNN requires laborious work of neural architecture tuning for various types of graph-structure data. Graph NAS mitigates strenuous tuning work via automatically identifying the optimal architecture for a given graph analytical problem. Technically, we tailor the search space and design an efficient controller to explore well-performing architecture for GNN.

2 From NSF Basic Research to Addressing a Market Opportunity

2.1 NSF Lineage

This proposed project develops from the NSF lineage work entitled *CAREER: Human-Centric Big Network Embedding, Award Number 1750074*. Network embedding is currently employed to learn a low-dimensional representation to facilitate network analytics applications, including node classification and network visualization in recommendation systems. This NSF lineage project has developed advancing tools of graph neural networks to improve the network embedding learning in various real-world applications, including social network and biochemical modular analysis. The lineage project also investigates a novel direction to explore how human beings could better understand the results. This multidisciplinary research's successful progress is currently leading to advances in enabling domain experts to interactively and quickly analyze big network data with human knowledge, thus positively impacting various information systems' online activity. **Amazon has been a critical partner in this project by providing us with real market-demands and feedback in our developments**.

Besides, the ongoing project's current results have helped develop a human-centric framework for modeling and incorporating human knowledge in network embedding, tackling data challenges in ML, and enabling interpretation and interaction of network embedding results. Our team has investigated multiview learning and deep structured frameworks to integrate three human knowledge types from the node-, edge- and community-level into a unified framework. Given that real-world online activity could contain heterogeneous, large-scale, and dynamic human knowledge, our research group has developed corresponding solutions to handle the problems. Our team also developed global and local interpretation algorithms to explain network embedding and interactive learning algorithms to integrate user feedback to facilitate the human understanding of our research results. **Our team has experienced the existing manufacturing problems from a practical and realistic perspective from our past collaborations with Samsung. We recognize the missing computational elements to bridge the gap between sophisticated recommendation systems and the technologies tailored to assist humans to take full advantage of advanced ML-tools in e-commerce**.

Based on our previous successful outcomes, we target to automate the network embedding learning by incorporating human expert knowledge in this proposed translational research. Due to the diverse data characteristics and modalities in real-world applications, the underlying analyzing tools, such as Graph Neural Networks, have to be laboriously and carefully designed. Our proposed framework plans to exploit the interpretation results and human knowledge to construct a robust and complete search space of analyzing tools, automatically optimizes the neural architectures, and improves network embedding learning by adapting to different scenarios.

2.2 Relevant NSF Lineage Results and Broader Impact

Our lineage project has already shown promising results reported in [1], where we presented Auto-GNN (AutoGNN), a Neural Architecture Search of Graph Neural Networks, to find the optimal neural architecture given a node classification task. Our team designed the search space, RCNAS controller, and constrained parameter sharing strategy explicitly for the message-passing-based GNN. Our experiment results show the discovered neural architectures achieve competitive performance on both transductive and inductive learning tasks. The proposed RCNAS controller searches the well-performed architectures more efficiently, and the shared weight could be effective in the offspring network under constraints.

One major application of the Auto-GNN project results is recommendation systems for e-commerce applications, as pictured in Fig. 1. Personalized recommendation systems are ubiqui-

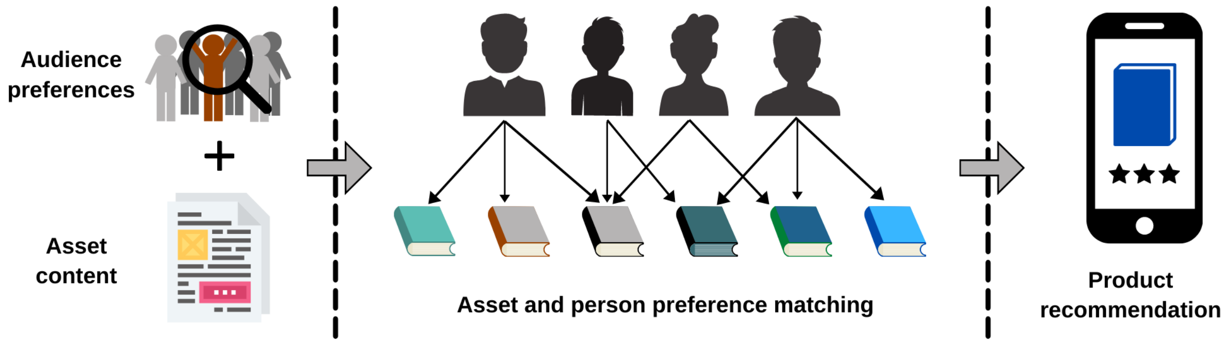

Figure 1: Auto-GNN application in e-commerce's recommendation systems.

tous and the primary source of revenue for many online services such as E-commerce, advertising, and social media. At its core, a recommendation system estimating how likely a user will adopt an item based on the historical interactions like purchases and clicks. As such, collaborative filtering (CF), which focuses on exploiting the past user-item interactions to achieve the prediction, remains to be a fundamental task towards the effective personalized recommendation. CF's most common paradigm is to learn latent features (a.k.a. embedding) to represent a user and an item and perform prediction based on the embedding vectors. Matrix factorization is an early model that directly projects the user's single ID to her embedding. Later on, several researchers found that augmenting user-ID with their interaction history as the input can improve embedding quality. Because of the user-item interaction graph, researchers could see these improvements from using the user's subgraph structure — more specifically, her one-hop neighbors — to improve the embedding learning. Inspired by this, there is a surge of works recently that data scientists have proposed to model on such graph-structured data for practical user profiling. The essential idea behind them is to represent each user or item as a subgraph and then capture this subgraph's structure information by exploiting powerful graph neural networks GNNs.

Although GNNs based recommendation systems have shown promising performance in various industrial applications, such as social media, e-commerce, and advertising, it remains challenging to successfully deploy GNNs-based systems in practice because of two significant reasons. First, the user's interests often change over time, and most of the customer's next behaviors are affected mainly by his/her recent actions, putting demands for model retraining frequently [2]. Second, the success of GNNs usually requires a lot of laborious works for architecture search. Thus, it is crucial to design an automated GNN framework to liberate people from these tedious works [3]. This project will provide a tailored neural architecture search solution for GNNs to tackle this problem. By automating the GNN training, we expect to easily accomplish an optimal GNN that captures users' real interests on time.

Besides, to provide more powerful GNN automatically, we can also build reliable relationships between items using fertile item attributes. Data scientists mainly estimate item interactions through two heuristic approaches—the first aims to compute item similarity based on their characteristic features using different distance metrics. Although intuitive and straightforward, such an approach may suffer from sparsity and missing values issues. Besides, it is also challenging to choose the effective distance metric. The second approach targets to estimate item relationships based on user behaviors. However, given that recommendation systems can only expose a small portion of items in a short period, this approach may face a severe cold-start problem. We propose Any2Graph to learn a reliable item to item relation graph from item attributes to tackle this problem. Our team can directly use the learned item relation graph to boost the performance of item-based recommendation tasks.

2.3 Market Analysis

Small businesses worldwide are currently looking to improve their online operations to serve a market shifting to buy mainly on the web, which is often an overlooked collateral effect of the current lookdowns and stay-at-home restrictions worldwide. Recommendation systems are a valuable tool in e-commerce for small businesses because they reduce data overload by providing meaningful results to customers that increase the possibility of converting a sale. Allowing small businesses to build and deploy robust recommendation systems becomes more economic important when knowing that only in the US, more than 47% of jobs come from small businesses.

Particularly, IBISWorld analysts indicate that the business analytics software industry will generate 86.7 billion US dollars in revenue in 2021 in the US; as shown in Fig. 2, the e-commerce market represents 17% of the market with a reported 11.3% annual growth over the past five years [4]. This industry heavily focuses on large enterprises with the data requirements and budget sizes necessary to take full advantage of recommendation systems, leaving an unmet need for smaller firms. **However, IBISWorld's report indicates that smaller businesses will become more critical to this industry over the next five years as they adopt enterprise technologies, that will eventually become the standard**. We see an opportunity to serve this market by developing easy-to-use ML tools with intuitive user interfaces tailored to retail and e-commerce managers with a basic programming background.

Business analysts from IBISWorld also indicate that online businesses are already heavy users of enterprise software. Online companies are exceptionally prepared to adopt enterprise software because they already heavily use information technology as a core part of their business. The e-commerce market segment is relatively mature due to its early adoption of enterprise software and will continue growing as online businesses displace traditional competitors. Altogether, online companies generated more than 4.0% of industry revenue in 2020 of the business analytics market [4].

On the one hand, we believe that our lineage project can reach a more mature commercialization potential with translational research that can build on basic scientific research. To further lower the complexity of the production of less expensive production of tailor-made recommendation systems, we propose to apply AutoGNN to discover architectures for more applications such as graph classification and link prediction. By implementing more advanced graph convolution techniques in the search space to facilitate neural architecture search in different applications. On the other hand, we have created a team specifically designed to bridge the gap between our fundamental research and to find a product-market fit, specifically by understanding what are the needs in the market, defining our value proposition, understanding and defining customers, and finding a repeatable and scalable business model.

2.4 Competitive technologies analysis

ML is a disrupting force across many industries; Statista projects ML-solutions' demand to grow more than 50% year to year over the next five years [5]. **Commercial recommendation technologies** such as Recombee [6], Crossing Minds [7], ExpertRec [8], and Strands [9], respectively, are increasingly investing in ML automation techniques. We are looking to complement these efforts by further expanding our human-centered explainability technologies that are already visible in our published research and our popular open-sourced projects. **We firmly believe that our competitive advantage is our strong focus on ML automation, which we inherently combine with interpretable artificial intelligence (AI) to maximize human knowledge with intelligent machine support**. Specifically, our work in Auto-GNN can be used in interpretation because of the inherent user-item connection of GNN by knowing how two items are correlated, i.e., when a user has bought an item or shown a strong interest in a product, human experts can use the visual

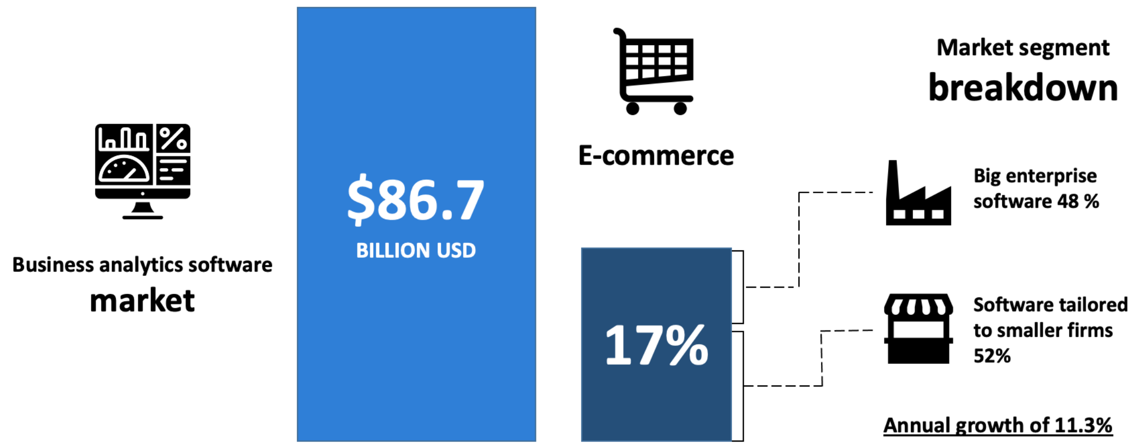

Figure 2: Business analytics software industry and e-commerce market segment concentration breakdown

representation of such correlations to improve their recommendation systems.

On the other hand, our indirect competitors also include the **Open Source** libraries Keras [10–12] and Tensor Flow [13–15]. While they offer elements to build recommendation systems, both libraries are not flexible for small businesses. Nevertheless, our goal is to push forward this effort by minimizing the business owners' learning curve and accelerating our recommendation system platform adoption. We do so by building proprietary software dedicated to ML automation on top of open-sourced neural architectures widely used by the AI community. Our vision is to use Automation and interpretation as key elements to make recommendation systems widely available for small businesses worldwide.

2.5 Intellectual property

Our team is protecting the intellectual property generated in this project under the open software license [16]. This license type will allow us to open-source our core system with basic functionalities to enable potential customers to build their recommendation system upon a trial version, which allow users to create a basic recommendation system, will be published with Apache License 2.0, allowing users to use it for any purpose. On the other hand, we plan to publish the advanced functions, such as automated machine learning, interpretable machine learning as well as state-of-the-art recommendation modules, under the GPL v3 license, which forbidden users to privately develop their product based on these modules and all of the developed software based on such modules must be open-source software as well. Our goal is to encourage academic researchers to conduct research based on our product while protecting our work from illegal business usage.

3 Technical Challenges and Applied Research Plan

This project's primary goal is to build effective models to serve various real-world applications, e.g., recommender systems, and time-series forecasting, via graph representation learning. Real-world data are often interlinked and can be formulated as graphs, consisting of nodes and edge links. For example, in social networks, friendship connections can be naturally abstracted as a graph, where each node is associated with a user, and each link is related to a friend connection [17–19]. In some other domains, the entities can be inherently connected. For example, a company's future sales can be both related to the company's historical sales and those of its competitors. A powerful tool to model these relationships is graph neural networks (GNNs), which

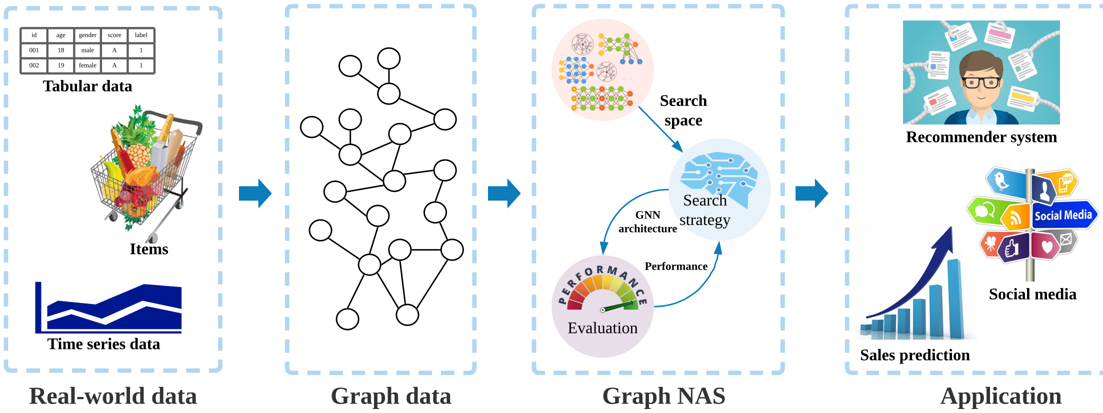

Figure 3: A schematic of automated graph neural network (AutoGNN) framework. We first transform the real-world data to graph data with Any2Graph module. We then enable end-to-end training with graph neural architecture search (Graph NAS) for various downstream applications.

have recently achieved great success in many graph analytical applications [20–22], at which the connected nodes with large weights are embedded similarly to benefit downstream tasks.

In this project, we propose an automated graph neural networks (AutoGNN) framework to model any type of data with graph representation learning (Figure 3). The project consists of three research objectives: an Any2Graph graph learning module that transforms any data (e.g., tabular data, images, and time-series) to a graph (Section 3.2), a graph neural architecture search (Graph NAS) module that discovers the optimal GNN architecture for different applications (Section 3.3), and the deployment of AutoGNN in real-world systems (Section 3.4).

3.1 Key Technical Challenges

Challenges for Transforming Data of Any Formats to Graph. It is non-trivial to learn the graph structure for realistic data due to the following challenges. First, it is unclear what feature entities should be regarded as the individual nodes to formulate the application-specific graph. For example, the different tabular data features interact in the recommendation system, while different time-series data samples should be interlinked in the prediction task. Second, it is not easy to tailor the graph structure to deal with the given application task well. For example, for click-through rate prediction tasks in a recommendation system, it is non-trivial to select the efficient feature interaction to reduce the ML system complexity and construct a more efficient ML system.

Challenges for Graph Neural Architecture Search. Porting the existing neural architecture search algorithms to find graph neural architectures is non-trivial due to the two significant challenges. First, the search space of GNN architectures is different from the ones in existing NAS literature [23]. Second, the traditional controller is inefficient to discover a well-performing GNN architecture [24]. Specifically, the graph convolutions in GNN architecture are specified by a sequence of actions, including aggregation, combination, and activation, and the representation learning capacity of GNN architecture varies significantly with the slight modifications of action.

Challenges for the Deployment of AutoGNN in Real-World Systems. Even though we can automate the process of graph construction and graph neural architecture design, it remains a challenging task to deploy AutoGNN in real-world systems due to two challenges as follows. First, the data formats in different applications can be quite different. We need a unified interface to adapt to different types of data. Second, real-world systems can be on a very large scale. Thus it is crucial to perform graph construction and graph neural architecture design efficiently.

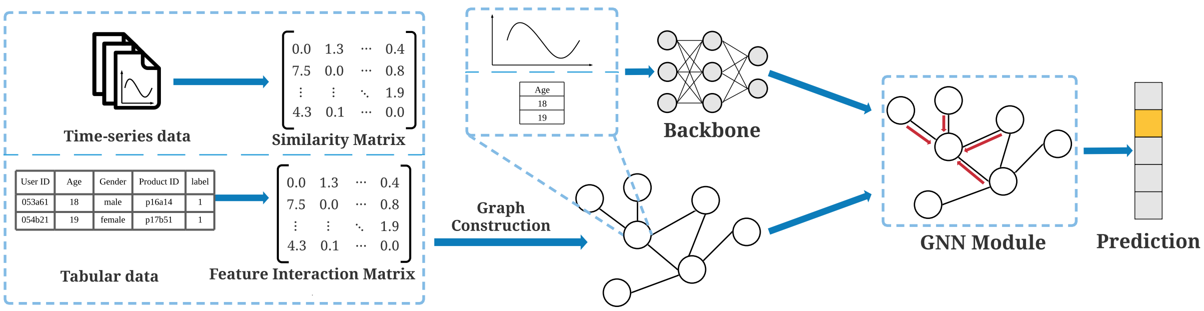

Figure 4: Graph learning for tabular and time-series data. The graph is constructed based on the tabular data's feature interactions or the time-series' pair-wise similarities. A backbone network extracts features for each instance, followed by a GNN module to aggregate the features with the constructed graph.

3.2 Learning Graph Structures with Any2Graph

This subsection introduces Any2Graph for transforming data of any given format to a graph structure (Figure 4). Specifically, we focus on tabular data and time-series data, which are very common in real-world e-commerce applications. Any2Graph can also be generally applied to other data formats such as images. The graph is learned based on the characteristics of the data formats: for tabular data, we consider the interactions of different features; for time-series data, we consider the dependencies of different time-series. The learned graph can then be combined with the powerful graph neural networks to learn the graph structure and the representations.

3.2.1 Problem Statement

The main goal of graph structure learning is to construct a graph from the data in any format. For the sake of simplicity, we focus on tabular data to describe our problem; the formulation can be easily extended to other data formats. Given a tabular dataset $\mathbf{x} = [x_1, \cdots, x_M]$, where M represents the number of interlinked features, we aim to predict a discrete label or continuous value y via building a mapping function $f(\mathbf{x})$. Specifically, the mapping function can be learned by minimizing the problem-specific loss, such as cross-entropy in binary classification. We aim to learn the underlying interactions with a graph and model function $f(\mathbf{x})$ with GNNs to capture the data relatedness explicitly. More concretely, our goal is to learn a graph adjacency matrix, where the element in the matrix represents the link weights between its corresponding nodes, such that the loss of the downstream tasks can be minimized to achieve superior performance.

3.2.2 Graph Learning for Tabular Data

Tabular data usually contains a mix of discrete and continuous feature column [25–27]. The patient risk prediction is an excellent example because the patient's health record is one of the typical tabular data. In the records, patients' age is a discrete feature column, and the gender of patients is a continuous feature column, where the task is to assess the risk of a patient in developing a target disease based on the records. Modeling the sophisticated interactions between feature columns plays a key role in making an accurate prediction [28, 29]. Many deep learning-based models have been proposed to learn the high-order interactions [30]. Specifically, they concatenate all the feature embeddings and feed them into special deep neural networks (DNN) like multilayer perceptron (MLP), where the high-order interactions are implicitly captured by the hidden units in the higher layers. However, these implicitly learned interactions cannot easily be extracted for interpreting the model decision because all feature columns are entangled together as soon as reaching the second hidden layer. Several existing works have also demonstrated that the model performance may benefit from the explicit modeling of feature interactions [26].

Instead of implicit interaction learning, we propose to extend the graph learning framework to a tabular domain for explicitly learning interactions between columns. Although tabular data does not have explicit feature relationships, these relationships can be learned to formulate the graph structure. Specifically, we could regard each feature column as an independent node in a graph and transform the task of modeling feature interactions to modeling link weights in graph adjacency structure [31]. Given the learned graph structure and a sample, we assign the series of feature values to the initial node attributes and run GNNs to update the node embeddings. The number of graph convolutional layers determines the order of feature interactions. Based on the updated node embedding, a pooling layer is applied on the node-set to generate the final graph embedding, capturing the sample's essential information for the prediction task. Thus, the interactions can be explicitly learned and extracted from the learned graph adjacency structures.

3.2.3 Graph Learning for Time-Series Data

Many forms of data can be formulated as time-series, e.g., temperature, electricity usage, exchange ratio, and stock price [32,33]. Time-Series analysis aims to understand and predict time-series with machine learning techniques [34]. For example, time-series forecasting is the task of predicting the next point's value based on the history data. It has various real-world applications, such as electricity usage prediction [35] and stock price prediction [36]. Another example is time-series classification [37], which aims at recognizing a time-series automatically, with various scenarios, such as signature verification [38] and person identification [39,40]. The recorded time-series data can be interlinked [41]. For example, the rise of temperature may affect electricity usage since people tend to increase air conditioner usage. Leveraging such information will help improve the prediction and classification accuracy of time-series data. However, in real-world applications, the relationships among the time-series can be very complicated [42,43]. A time-series can be related to several other time-series [43,44]. For example, a company's stock price can be simultaneously impacted by its revenues and its competitors' statistics. A machine learning model needs to make predictions based on the current series's features and its connections to other series. Motivated by this, we aim to exploit the relationships among different time-series by graph learning. Specifically, we regard each time-series as a node of a graph and connect the related time-series with links. We connect the time-series with similarity metric learning. The learned graph can then be used in GNN for downstream applications, such as time-series prediction and time-series classification.

3.3 Graph Neural Architecture Search

Given the input graph structure, GNN learns the sophisticated node interaction and updates the node embeddings, which could be applied to solve a downstream task. However, the success of GNN is accompanied by laborious work of neural architecture tuning, aiming to adapt GNN to different types of graph-structured data [20, 21]. For example, attention heads in the graph attention networks [45] have to be selected carefully for citation networks and biological networks, respectively. GraphSAGE [46] has been shown to be sensitive to the dimension of hidden units. These handcrafted architectures require an extensive search in the design space through many trials and tend to obtain suboptimal performance when they are transferred to other graph-structured data. Based on the observations, in this research objective, we investigate the graph neural architecture search (NAS) to automatically identify an optimal architecture for any given graph analytics problem. Figure 5 shows an overview of our graph NAS mainly consisted of two components: search space and search controller. The search space contains a large number of GNN architecture variants and covers the state-of-the-art models in the literature. The search controller explores the search space and preserves the best neural architecture at each epoch while applying reinforcement learning to update the search progress towards better GNNs.

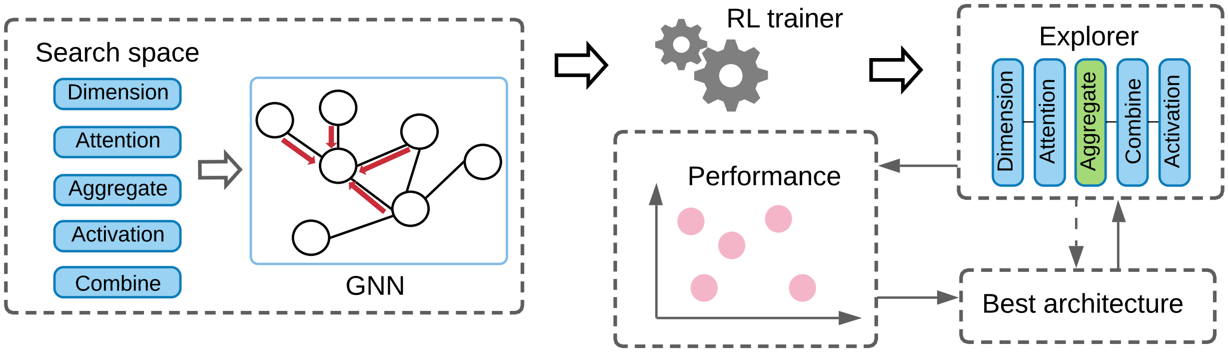

Figure 5: An overview of graph NAS. The search space includes a large number of GNN variants. The search controller preserves the best architecture found so far and explores the search space by modifying only one neural class (e.g., aggregate) of the preserved architecture. Based on the new architecture's performance, the RL trainer updates the search algorithm towards exploring better GNNs.

3.3.1 Problem Statement

The main goal of graph NAS is to identify an optimal graph neural architecture to maximize model performance for any a given graph analysis task. Formally, let $\mathcal{F}$ denote the search space including a large number of graph neural architecture variants, and let M denote the performance metric to evaluate the GNN model. We aim to find the optimal GNN architecture $f^* \in \mathcal{F}$ accompanied with the best metric M^* on the validation dataset. In the graph analysis problem of node classification, metric M could be represented by F1 score or area under the curve (AUC).

3.3.2 Search Space of GNN Architectures

We propose to construct a general search space of GNN architectures composed of layers of message-passing based graph convolutions. Formally, in the k-th layer, graph convolution could be expressed as:

$$\begin{aligned} h_i^{(k)} &= \text{AGGREGATE}(\{a_{ij}^{(k)} W^{(k)} x_j^{(k-1)} : j \in \mathcal{N}(i)\}), \\ x_i^{(k)} &= \text{ACT}(\text{COMBINE}(W^{(k)} x_i^{(k-1)}, h_i^{(k)})). \end{aligned} \tag{1}$$

$x_i^{(k)} \in \mathbb{R}^{d^{(k)} \times 1}$ denotes the embedding of node i at the k-th layer. $d^{(k)}$ is dimension length in layer k. $\mathcal{N}(i)$ denotes the set of neighbors adjacent to node i. $W^{(k)}$ denotes the trainable matrix used to project embedding into new space. $a_{ij}^{(k)}$ denotes the relation coefficient between nodes i and j, which is computed from an attention function. Function AGGREGATE is applied to aggregate neighbor representations and prepare intermediate embedding $h_i^{(k)}$. Besides, the function COMBINE is used to combine information from the node itself as well as embedding $h_i^{(k)}$, and function ACT is used to activate the node embedding. Based on Equation (1), each layer of graph convolutions is determined by the following 5 neural classes: dimension length, attention function, aggregate function, combine function, and activation function.

Given the above five neural classes, a GNN architecture could be specified by a string of length $5n$, where n denotes the number of graph convolutional layers. The search space contains all the variants of such architecture string. To optimize performance for any a given task, the search space has to be huge enough to support the identification of the optimal GNN architecture. We provide a candidate set for each neural class accompanied with the cardinality of c, via collecting a series of existing state-of-the-art functions. The search space size is given by c^{5n}, which exponentially increases with c and n.

3.3.3 Graph Neural Architecture Search Controller

The learning capability of a GNN architecture varies significantly with slight modifications of some architecture components, which makes the neural architecture search hard work. Taking the neural class of aggregate function as an example, the node classification accuracy will be improved by only replacing the max pooling with summation [24]. The search controller must learn about which part of architecture modifications contributes more or less to the performance improvement to accelerate the search progress. To search GNN architectures efficiently, we propose a new controller named reinforced conservative neural architecture search (RCNAS), as shown in Figure 5. It consists of three key components: a conservative exploiter to preserve the best architecture, a small-step explorer to search the next architecture with slight modifications on the preserved best architecture, and a reinforcement learning trainer to update the search controller based on the searched model's performance.

The conservative exploiter is applied to maintain the best neural architecture found so far. In this way, the following architecture modification is performed based on a reliable parent, which ensures fast exploitation towards the better offsprings in the vast search space. If the offspring GNN outperforms its parent, we will update the best neural architecture; otherwise, the best one will be kept and reused to generate the next offspring GNN.

The small-step explorer is proposed to modify the best architecture found so far by choosing a neural class that waits for exploration. Based on the selected class, the controller generates the detailed functions to modify the best architecture. As shown in Figure 5, the neural class of aggregate is chosen, while all the aggregate functions in the preserved best architecture will be modified to generate the offspring GNN. The controller learns the corresponding model performance variation to identify the powerful aggregation functions through such slight modifications.

The reinforcement learning trainer uses REINFORCE rule of policy gradient [47] to update the controller. By comparing with the preserved architecture, the controller is trained towards generating functions of the selected neural class if the new architecture is superior. Otherwise, the controller tends to forget the selected functions in the future neural architecture search.

3.4 Innovations and Evaluations in Real Systems

This research objective is to investigate how AutoGNN could be embedded in real systems. We will first focus on some representative tasks, including recommender system, time-series analysis, and interaction detection. The PIs have collaborated on publications and established state-of-the-art methods for all the tasks mentioned above. Such research experiences help demonstrate the applicability and effectiveness of the proposed algorithms in real applications.

3.4.1 Recommender System

Graph-based recommender systems have attracted significant attention recently. The core idea behind this is to describe the complex relationships between users and items via a bipartite graph. With a well-defined user-item interaction graph [31], we can accurately infer a user's next preference based on its interactions and the similar-minded users explored over the graph. For example, if two users buy many of the same products in a time, the interaction graph can preserve their common interests by allowing them to be connected via several "user->item->user" meta-paths. Graph neural networks are inherently effective in modeling bipartite graphs since they can recursively aggregate informative messages from neighboring nodes. As a result, a target user could easily reach out to other users' preferences by using a two-layer GNNs. However, it remains challenging to successfully deploy GNNs-based systems in practice for two significant reasons. First, user interests often change over time, and most of the customer's next behaviors are largely affected by his/her recent behaviors, putting demands for model retraining frequently [2]. Second, the success of GNNs usually requires a lot of laborious works for architecture search. Thus, it is important to design an automated GNN framework to liberate people from these tedious works [3]. This project will provide a tailored neural architecture search solution for GNNs to tackle this problem. By automating the GNN training, an optimal GNN that captures users' real interests on time could be easily accomplished.

In addition to providing more powerful GNN automatically, we can also build reliable relationships between items using fertile item attributes. Item interactions are currently mainly estimated via two heuristic approaches—the first aims to compute item similarity based on their features using different distance metrics. Although intuitive and straightforward, such an approach may suffer from sparsity and missing values issues. Besides, it is also challenging to choose the effective distance metric. The second approach targets to estimate item relationships based on user behaviors. Given that only a small portion of items can be exposed in daily life, this approach may face severe cold-start problems. We propose Any2Graph to learn a reliable item to item relation graph from item attributes to tackle this problem. The learned item relation graph can be directly used to boost the performance of item-based recommendation tasks.

3.4.2 Time-Series Analysis

We use concrete examples to discuss the benefits of using graph and graph neural networks to model the relationships among multiple time-series. Here, we take multi-variate time-series forecasting as an example. We note that our graph modeling method is general and applicable to other time-series analysis tasks, such as time-series classification.

We consider the consumption forecasting task using historical data to inform decision making in e-commerce. For example, we aim to predict Visa Inc's consumption, which can be naturally formulated as a time-series, where each point implies the consumption for a specific month. Given the consumption in the previous months, our objective is to predict the consumption in the next month. Traditional methods make predictions for each time-series separately. However, these methods will have sub-optimal accuracy, measured by the mean square error of the predictions on the true values. This is because they do not incorporate related yet independent historical data. For example, the user with similar income may have similar consumption. Thus, it is important to model these relationships to improve performance and graph learning can tackle this problem. by using an adjacency matrix to model the pair-wise similarity of these time-series. This matrix can be obtained by comparing the time-series pair-wisely or by the income level. With this matrix, we can employ GNN to incorporate this similarity information into a unified learning framework. By modeling the relationships among the time-series, we can achieve better accuracy in forecasting.

The applicability of graph learning of time-series is far beyond the above example. It can be applied to time-series in various domains, such as stock price and exchange ratio [33, 48]. It can also be used in other tasks such as classification [49] and clustering [50,51]. Our goal is to provide a general graph learning framework for time-series to benefit all the downstream analysis tasks.

3.4.3 Interaction Detection

Learning feature interactions is crucial for many prediction tasks to achieve high-performance [28, 52, 53]. For example, it is reasonable to recommend a user to use Lyft on a rainy day at off-work hours (e.g., during 5-6 pm). In this situation, considering the feature interaction *rainy* and *5-6pm* is more effective than considering the two features separately. Previously, most models focus on either manually designing feature interactions or simply enumerating all possible feature interactions [54]. However, these methods will have sub-optimal accuracy or even worse performance since useless interactions may introduce noise and complicate the training process. Some recent work considers each feature interaction's importance through the attention mechanism. However, these methods still take all feature interactions into account. Moreover, they capture each feature interaction's contribution to the prediction task, failing to capture the holistic contribution of a set of feature interactions together. The proposed graph learning technique in this project is helpful to tackle this problem. Specifically, Any2Graph targets to learn a pair-wise adjacent matrix to indicate the importance of each feature interaction. With this matrix in hand, feature interactions beyond second-order can also be modeled along the edge over the graph. We can employ a graph neural network to explicitly capture multi-hops of feature interactions for joint optimization to achieve this goal. After the training, we can use the resultant adjacent matrix to provide feature-level explanations for the prediction.

4 Achieving Societal Impact through the Realization of Commercial Potential

4.1 Commercialization strategy

It is essential to highlight that we have formed a team to push forward our research and development efforts, and commercialize our technology. We are actively seeking non-dilutive funding support from federal agencies such as NSF and DoD agencies. NSF has been a critical supporter of our research and developments. We believe that our successful results in the CAREER: Human-Centric Big Network Embedding grant prepared us well to propose this first attempt to commercialize our research. This partnership is essential for our commercialization strategy as it can also extend to other programs dedicated to pushing lab technologies into the market, as is the case of Small Business Innovation Research grants. Similarly, other important partners are DoD agencies that have seen potential in our packages, as it is the Department of Defense's particular case, which has been a sponsor of many of our AutoML technologies.

In addition, note that the technology commercialization office and Startup Aggieland, the on-campus incubator at Texas A&M University, will also be valuable assets in our pursuit of technology commercialization. Mainly in two areas, their mentorship and entrepreneurial curriculum will be invaluable for our team, and secondly, they already possess a strong network of angel investors and domain experts. Note that Texas A&M University also counts with its retailing studies department, which directly studies the industry we want to innovate in.

4.2 Assessment plan to gauge the success of the research partnerships and third-party collaboration

We want to promote entrepreneurship as a career path for our graduate students and postdoctoral fellows. An effective way to do so is through participation in business competitions. Two central motivations to encourage our teams' participation at Business Competitions are to push us to think outside the box, refine our business model, and identify mentors who can help us raise funds. We believe that business competition is an excellent and practical way to gauge our research commercialization's success and our partnerships alike in potential investors' eyes.

We currently measure the success of our packages depending on the number of downloads, stars and forks in our GitHub repository. We believe that to start a business, we first need to create a community. The open-source community has also been an essential portion of our commercialization efforts. Notably, their demand, measure through downloads and reviews, has historically correlated to useful package features to the community. Releasing open-source packages serve two goals: first, it helps us gauge and understand download activity and user-feedback reviews to measure the success of our packages, and second, it is also a great way to build a community of programmers, researchers, and more importantly, other entrepreneurs that want to commercialize technologies in recommendation systems.

The NSF I-corps program will also help us gauge the success of our business partnerships. At this point, one of the business hypothesis we need to validate or invalidate is that our business model will follow conventional practices in the business intelligence industry, divided into non-recurring and recurring revenue streams. **Non-recurring revenue streams** involve setup infrastructure costs such as AI & ML model building, platform installation, operator training, deployment expenses, and data preprocessing. Our **recurring revenue streams** may follow standard models of cloud container companies [55] such as AWS®. They include ML deployment operation, license fees, data volume analyzed, and extra premium features related to security and data encryption. We also believe that our software as a service (SaaS) expected margin is 80 to 85%, and our projected monthly burn rate is expected to be $40,000 for the first year. The I-corps program and their business model canvas methodology can undoubtedly help us gauge the validity of our value proposition and product-market fit in our quest to translate academic research and technologies into commercial use rapidly.

5 Project Team

This team comprises expertise from data mining and machine learning systems (Hu), data analytics, entrepreneurship and technology commercialization (Costilla-Reyes), machine learning (Zha), including faculty and graduate students from the Data Analytics at Texas A&M (DATA) Lab.

PI Dr. Xia "Ben" Hu is an Associate Professor and Lynn ’84 and Bill Crane ’83 Faculty Fellow at Texas A&M University in the Department of Computer Science and Engineering. He has published more than 100 papers in major data mining venues. His articles have received seven Best Paper Award (candidate), and he is the recipient of the JP Morgan AI Faculty Award, the Adobe Data Science Award, and the NSF CAREER Award. An open-source package developed by his group, namely AutoKeras, has become the most used automated deep learning system on Github (with over 7,600 stars and 1,200 forks). Dr. Hu’s work on deep collaborative filtering, anomaly detection, and knowledge graphs is part of the TensorFlow package, Apple production system, and Bing production system. Hu’s work has been cited more than 7,000 times, with an h-index of 38. He was the conference General Co-Chair for WSDM 2020.

Dr. Alfredo Costilla-Reyes, is currently a postdoctoral researcher at Texas A&M University in the Department of Computer Science and Engineering. He graduated from Entrepreneurship and Technology Commercialization program at Mays Business School and the doctorate program in Electrical Engineering, both from Texas A&M University. Dr. Alfredo has been a recipient of the NSF I-Corps Site, 2017-2018 Kirchner, Silicon Labs and the McFerrin-Entrepreneurship Fellowships, and the prestigious Mexico National Youth Award, presented by the president of Mexico for his contributions in science, technology, and entrepreneurship. Specifically, Dr. Alfredo has led projects regarding embedded software and systems for future agriculture, battery-less wearable consumer electronics, application-specific integrated circuits, and wireless systems for IoT applications. His research and entrepreneurial endeavors have participated in YCombinator’s YC120 event, Silicon Valley Bank Trek, McFerrin Center for Entrepreneurship business incubator, and Rice University’s OwlSpark accelerator.

Daochen "Frank" Zha is a Ph.D. student from the Department of Computer Science and Engineering at Texas A&M University. His research centers on machine learning and data mining, particularly in optimizing solutions with reinforcement learning. He is leading the RLCard project, an open-source platform for reinforcement learning in card games. He is also the major contributor to several other machine learning open-source projects, including PyODDS, an end-to-end system for outlier detection, and TODS, an AutoML framework for time series outlier detection. The projects developed by Mr. Zha have attracted more than 1,000 stars in total on Github.

The team has established long-term collaboration, co-authored publications, and worked together on active DARPA and NSF projects to conduct fundamental research in developing interpretable and automated machine learning systems. Our past collaborations have provided a solid ground to potentiate the technology commercialization of our NSF lineage technologies.

6 Partnerships

Outcomes of the proposed project will facilitate automated recommender system in e-commerce applications, by providing a more effective ranked list of items/products automatically. To better understand user needs and commercialization opportunities, as well as evaluate the effectiveness and efficiency of the developed systems on real-world datasets, PI Hu and the team will actively collaborate with corresponding domain experts from leading companies, including Adobe, Apple, Amazon and Samsung, through funded joint research and development. Amazon and Samsung's support is critical in our quest to pursue our current technologies' commercialization (see letters of support). Notably, as the leading e-commerce, Amazon is very well positioned to push forward our efforts, and their experience with real-world problems will help us deliver better solutions tailored to small businesses around the globe. Moreover, we have collaborated extensively with Samsung's Mobile Ads AI team, which has provided vital specific problem needs, collaborative data curation, evaluation platforms, performance metrics, and feedback. Both partnerships are key for our commercialization strategy moving forward.

The team has collaborated on publications or research products with each of the aforementioned entities and established methods for data-sharing through direct access or proxy via synthetic datasets with the companies. The purpose of such a broad-range of collaborators and partnership is to demonstrate the applicability and effectiveness of the proposed algorithms on real-world datasets, as well as solve problems that matter to business owners.

7 Training Future Leaders in Innovation and Entrepreneurship

We would like to take this proposal's opportunity to empower our team, that includes faculty members, postdoctoral researcher and PhD students, by harnessing the opportunity to learn and grow our entrepreneurial spirit. This proposal's I-corps component will help our graduate students and postdoctoral researcher serve as the Entrepreneurial Lead and find this project's commercialization. Such activities heavily stress the importance of reaching out to industry experts and getting their opinion about their current needs. As it is true in science, defining the correct business problem is more important than starting working in a solution based solely on a groundless hypothesis. We also plan to train the team with Disciplined Entrepreneurship: 24 Steps to a Successful Startup [56] developed at MIT, consisting of a series of lessons and methodology to formulate and test business assumptions. Another implementation we will put in place in this project is The Startup Owner's Manual [57], an Step-By-Step Guide for Building a Great Company developed by Steve Blank, which is a method highly recognized for its replicable product-market fit testing process.

This PFI project will also help us learn and implement a lean methodology into our daily tasks. Specifically, we want to take advantage of the I-Corps training to validate and invalidate our hypothesis in a more systematic way. At the same time, this PFI opportunity is already encouraging to look for partnerships beyond academia into the industry and other startup-tailored ecosystems. An additional leadership development goal is to help our graduate student and postdoctoral researchers properly articulate a very technical idea into solutions that can find the support of investors. We incorporated in this proposal a path for students and researchers to participate in entrepreneurial events, ranging from local business competitions at Texas A&M University to internationally recognized startup accelerators such as YCombinator.

Naturally, a desirable outcome is to develop a product that our team can commercialize. However, an equally desirable outcome is to educate our future generation of scientists on ways they should formulate their grants to be commercialized. Another interesting philosophy in many entrepreneurial curricula, which we often fail to implement in academia, is to accept failure as a way to grow. In other words, we are going to teach our students and postdoctoral researchers to fail fast, in a way to invalidate flawed assumptions as soon as possible to focus on ideas that have the highest potential. An important aspect we want to implement with this grant is to leverage software development flexibility. Specifically, we want to exploit the ability to fast prototype solutions that we may think are important in the market, then assess how well or poorly receive such updates are for our users and then iterate product development again, creating in this way a lean development ideology and shaping our solutions according to real market needs.

8 Broadening Participation

This proposal's inclusion plans extend beyond the research team and commercial partners to use our developments to serve small businesses. As we explained in this document, the current stay-at-home restrictions worldwide have made visible the importance of e-commerce in the global economy. An equally important issue is that while large retailers are quickly adopting ML-based recommendation systems, these advanced tools are still costly and complicated, limiting smaller business owners' ability to benefit from their data. This project's mission is to close the currently growing digital divide between small businesses and large enterprises by providing access to high-quality and affordable ML tools for small businesses and be part of the vision of a more inclusive digital world for small business owners. We want to provide more accessible ML and help women, minorities, and persons with disabilities grow their online businesses and thrive in the digital era in our quest to build an equal world. In addition, the team will actively encourage female and undergraduate participation in our project through the definition of self-contained, manageable research 'units' that could be completed in one semester or over a summer.

In the same way, we have the goal to use AI ethically across all our developments. Recommenders are essential online channels: they shape the media we consume and the products we seek at any level small or enterprise. And yet, recommenders may be subject to algorithmic bias that can lead to negative consequences in recommendations offers. For example, job recommenders can target women with lower-paying jobs than equally-qualified men. News recommenders can favor particular political ideologies over others. And in our specific case, ad recommenders can exhibit racial discrimination. Overcoming such algorithmic bias is a top priority for our team and the reason why we have developed and implemented fairness-aware algorithms that we designed to maintain quality while dramatically improving fairness in recommendation systems [58]. Specifically, two key technical aspects of our efforts are implementing a sensitive latent factor matrix for isolating sensitive features and developing a novel sensitive information regularizer that extracts sensitive information that can bias other latent factors. Our vision is to serve as an example and help drive a more ethical use of AI in e-commerce.

References

[1] Kaixiong Zhou, Qingquan Song, Xiao Huang, and Xia Hu. Auto-gnn: Neural architecture search of graph neural networks. *CoRR*, abs/1909.03184, 2019.

[2] Yufei Feng, Fuyu Lv, Weichen Shen, Menghan Wang, Fei Sun, Yu Zhu, and Keping Yang. Deep session interest network for click-through rate prediction. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pages 2301–2307, 2019.

[3] Thomas Elsken, Jan Hendrik Metzen, Frank Hutter, et al. Neural architecture search: A survey. *J. Mach. Learn. Res.*, 20(55):1–21, 2019.

[4] Dan Cook. *IBISWorld US Industry (NAICS) Report 51121C. Business Analytics Enterprise Software Publishing in the US*, July 2020. Retrieved from IBISWorld database.

[5] Tractica. *Revenues from the artificial intelligence software market worldwide from 2018 to 2025, by region (in billion U.S. dollars) [Graph]*, April 22, 2019 (Accessed October 6, 2020). https://www.statista.com/statistics/721747/worldwide-artificial-intelligence-market-by-region/.

[6] PitchBook Data. *Recombee | Private Company Profile*, February 2021. Retrieved from PitchBook database.

[7] PitchBook Data. *Crossing Minds | Private Company Profile*, February 2021. Retrieved from PitchBook database.

[8] PitchBook Data. *ExpertRec | Private Company Profile*, February 2021. Retrieved from PitchBook database.

[9] PitchBook Data. *Strands | Private Company Profile*, February 2021. Retrieved from PitchBook database.

[10] Nikhil Ketkar. Introduction to keras. In *Deep learning with Python*, pages 97–111. Springer, 2017.

[11] Antonio Gulli and Sujit Pal. *Deep learning with Keras*. Packt Publishing Ltd, 2017.

[12] Jojo Moolayil, Jojo Moolayil, and Suresh John. *Learn Keras for Deep Neural Networks*. Springer, 2019.

[13] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016.

[14] Nishant Shukla. *Machine learning with TensorFlow*. Manning Publications Co., 2018.

[15] Joshua V Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, and Rif A Saurous. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, 2017.

[16] Lawrence Rosen. *Open source licensing*, volume 692. 2005.

[17] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation. In *The World Wide Web Conference*, pages 417–426, 2019.

[18] Liyu Gong and Qiang Cheng. Exploiting edge features for graph neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9211–9219, 2019.

[19] Yao Ma, Ziyi Guo, Zhaocun Ren, Jiliang Tang, and Dawei Yin. Streaming graph neural networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 719–728, 2020.

[20] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864, 2016.

[21] 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.

[22] Marinka Zitnik and Jure Leskovec. Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, 33(14):i190–i198, 2017.

[23] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[24] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? 2018.

[25] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10, 2016.

[26] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: a factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 1725–1731, 2017.

[27] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. Field-aware factorization machines for ctr prediction. In *Proceedings of the 10th ACM conference on recommender systems*, pages 43–50, 2016.

[28] Daria Sorokina, Rich Caruana, Mirek Riedewald, and Daniel Fink. Detecting statistical interactions with additive groves of trees. In *Proceedings of the 25th international conference on Machine learning*, pages 1000–1007, 2008.

[29] Ronald Aylmer Fisher. Statistical methods for research workers. In *Breakthroughs in statistics*, pages 66–70. Springer, 1992.

[30] Christoph Hofer, Roland Kwitt, Marc Niethammer, and Andreas Uhl. Deep learning with topological signatures. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1633–1643, 2017.

[31] Zekun Li, Zeyu Cui, Shu Wu, Xiaoyu Zhang, and Liang Wang. Fi-gnn: Modeling feature interactions via graph neural networks for ctr prediction. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 539–548, 2019.

[32] Steven Craig Hillmer and George C Tiao. An arima-model-based approach to seasonal adjustment. *Journal of the American Statistical Association*, 77(377):63–70, 1982.

[33] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 753–763, 2020.

[34] David E Goggin. Method and apparatus for time series graph display, November 6 2007. US Patent 7,292,245.

[35] K Muralitharan, Rathinasamy Sakthivel, and R Vishnuvarthan. Neural network based optimization approach for energy demand prediction in smart grid. *Neurocomputing*, 273:199–208, 2018.

[36] Nesreen K Ahmed, Amir F Atiya, Neamat El Gayar, and Hisham El-Shishiny. An empirical comparison of machine learning models for time series forecasting. *Econometric Reviews*, 29(5-6):594–621, 2010.

[37] Ben D Fulcher and Nick S Jones. Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):3026–3037, 2014.

[38] Paul Maergner, Vinaychandran Pondenkandath, Michele Alberti, Marcus Liwicki, Kaspar Riesen, Rolf Ingold, and Andreas Fischer. Combining graph edit distance and triplet networks for offline signature verification. *Pattern Recognition Letters*, 125:527–533, 2019.

[39] Yaoyu Li, Hantao Yao, Lingyu Duan, Hanxing Yao, and Changsheng Xu. Adaptive feature fusion via graph neural network for person re-identification. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2115–2123, 2019.

[40] Yantao Shen, Hongsheng Li, Shuai Yi, Dapeng Chen, and Xiaogang Wang. Person re-identification with deep similarity-guided graph neural network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 486–504, 2018.

[41] Prasanna Desikan and Jaideep Srivastava. Time series analysis and forecasting methods for temporal mining of interlinked documents. *Department of Computer Science, University of Minnesota, www-users. cs. umn. edu/˜ desikan/publications/TimeSeries. doc, accessed on January*, 26, 2014.

[42] Mengyuan Chen, Jiang Zhang, Zhang Zhang, Lun Du, Qiao Hu, Shuo Wang, and Jiaqi Zhu. Inference for network structure and dynamics from time series data via graph neural network. *arXiv preprint arXiv:2001.06576*, 2020.

[43] Slawek Smyl. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1):75–85, 2020.

[44] Michael D Morse and Jignesh M Patel. An efficient and accurate method for evaluating time series similarity. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 569–580, 2007.

[45] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[46] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. *arXiv preprint arXiv:1706.02216*, 2017.

[47] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pages 1057–1063. Citeseer, 1999.

[48] Bing Yu, Haoteng Yin, and Zhanxing Zhu. St-unet: A spatio-temporal u-network for graph-structured time series modeling. *arXiv preprint arXiv:1903.05631*, 2019.

[49] Ali Jalali and Sujay Sanghavi. Learning the dependence graph of time series with latent factors. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pages 619–626, 2012.

[50] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

[51] Hesam Izakian and Witold Pedrycz. Anomaly detection and characterization in spatial time series data: A cluster-centric approach. *IEEE Transactions on Fuzzy Systems*, 22(6):1612–1624, 2014.

[52] Lei Mei, Pengjie Ren, Zhumin Chen, Liqiang Nie, Jun Ma, and Jian-Yun Nie. An attentive interaction network for context-aware recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*, pages 157–166, 2018.

[53] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 3119–3125, 2017.

[54] Michael Tsang, Hanpeng Liu, Sanjay Purushotham, Pavankumar Murali, and Yan Liu. Neural interaction transparency (nit): Disentangling learned interactions for improved interpretability. In *NeurIPS*, pages 5809–5818, 2018.

[55] Donna L Hoffman and Thomas P Novak. A conceptual framework for considering web-based business models and potential revenue streams. *International Journal of Marketing Education*, 1(1):7–34, 2005.

[56] Bill Aulet. *Disciplined entrepreneurship: 24 steps to a successful startup*. John Wiley & Sons, 2013.

[57] Steve Blank and Bob Dorf. *The startup owner's manual: The step-by-step guide for building a great company*. John Wiley & Sons, 2020.

[58] Ziwei Zhu, Xia Hu, and James Caverlee. Fairness-aware tensor-based recommendation. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 1153–1162. ACM, 2018.