

Invention

Disclosure

Form



**Instructions**

* Provide as much detail as possible. Attach additional explanatory documents or materials (e.g. technical specifications, drawings, flowcharts, manuals, PPT presentations).
* Timing is critical to patent protection.

***Talk to IP Legal Before Making a Public Disclosure.***  - If the invention is patentable, premature publication, commercial activity or other non-confidential disclosure may damage or preclude patent protection for your invention in many countries, including the United States.

***Disclose As Soon As The Invention Is Clearly Conceptualized.*** - You do not need to wait until the invention is completely developed to seek patent protection. Under the “First to File” patent system in the U.S., filing as early as possible is preferable, even critical, to obtaining effective patent protection. This is especially true where others are filing patent applications in the same or adjacent technology areas.

* Questions? Contact Eva Wood in IPLegal (eva.wood@uhg.com or 952-205-6063).

**Send completed form and all attachments to: IPLegal@uhg.com**

**Cover Sheet Information**

|  |  |
| --- | --- |
| Title of Invention: | Node2Vec Embeddings for Patient Similarity V3 |
| Date Submitted: | August 2020, with PRB requested updates Oct 1st and Oct 7th |
| Segment funding development to-date: | none |

**Invention Technology Domain** (check most relevant)

🞏 AI / Machine Learning / Deep Learning 🞏 Genomics 🞏 Healthcare Processes

🞏 Cybersecurity 🞏 Blockchain 🞏 Internet of Things (IOT)

🞏 General Information Technology / Other

**Current stage of invention:**

🞏 Concept 🞏 Research Tool 🞏 Prototype

🞏 Beta Test 🞏 Ready to Deploy/License

**Keywords:** (List at least 5 keywords that describe the invention for patent searching and classification)

1. Patient Similarity 3. Embeddings 5. Machine Learning

2. Cosine Similarity 4. Node2Vec *(Add keywords, if applicable.)*

**Contributor Information**

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | | Alex Li | |
| Business Unit: | | Advanced Technology Collaborative | |
| 🕿 | 423-999-6818 | @ | [alex.li6@optum.com](mailto:alex.li6@optum.com), lialex111@yahoo.com |
| Current Residential Address: | | 17004 77th Ave N, Maple Grove, MN 55311 | |
| Work Location: | | 11000 Optum Circle, Eden Prairie, MN 55344 | |
| Country of Citizenship: | | United States of America | |
| Role in Conception: | | Designed and Implemented Invention | |

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | | Dan McCreary | |
| Business Unit: | | Advanced Technology Collaborative | |
| 🕿 | 612-986-1552 | @ | Dan.mccreary@optum.com |
| Current Residential Address: | | 3007 Cavell Ave. So. | |
| Work Location: | | 13625 Technology Dr, Eden Prairie, MN 55344 | |
| Country of Citizenship: | | United States of America | |
| Role in Conception: | | Provided technical advice and connected Alex Li with others in the field | |

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | | Parker Erickson | |
| Business Unit: | |  | |
| 🕿 |  | @ |  |
| Current Residential Address: | |  | |
| Work Location: | |  | |
| Country of Citizenship: | | United States of America | |
| Role in Conception: | | Provided technical advice | |

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | | Mark Megerian | |
| Business Unit: | |  | |
| 🕿 |  | @ |  |
| Current Residential Address: | |  | |
| Work Location: | |  | |
| Country of Citizenship: | | United States of America | |
| Role in Conception: | | Provided data structure and patient data | |

*Add additional boxes for each individual that may have contributed to the conception of the invention.*

*These individuals may later be considered inventors of the disclosed invention;*

*IPLegal will provide guidance to determine final inventorship.*

*Please place an asterisk “\*” next to the individual who should be considered the primary contact.*

**Invention Description**

**1.1 Problem solved and solutions/drawbacks to existing solutions**

The goal of the invention is to calculate the most similar patients to a target patient in a quick but accurate manner. The most similar patients retrieved can be used in care plan predication at the speed of patient encounter. There are no other existing solutions, but the algorithms used in the project has been applied to cases outside of the clinical setting.

This concept is innovative because it is the first project that has shown the feasibility of doing real-time patient similarity at the point of care of a patient encounter. Patient similarity can be done in parallel computation to greatly reduce time for similarity calculations. This invention shows that combination of machine learning on a graph representation and FPGA hardware makes this solution completely distinct in the healthcare space.

**1.2 Description of Invention**

The broad overview of the procedure is to have patient data stored in a graph database such as TigerGraph. Once stored, perform random walks on the data to obtain several paths of patients connected to each other. The random walks can be used as input to a Word2Vec model to find embeddings for each patient. The embeddings received from the model can be used with cosine similarity to detect which patients are similar. To further reduce the time for cosine similarity calculations, the embeddings can be used in a FPGA. FPGA allows fast parallel computations which greatly reduces the cosine similarity calculations on the embeddings. Not only does the embeddings allow less memory space due to the shrinking of dimensions, the FPGA allows faster calculations performed. The process of using random walks as input to a Word2Vec model is called Node2Vec. A more in-depth explanation on how this project is implemented will be discussed below.

A screenshot of a cell phone

Description automatically generated

**Figure 1. Architecture of the Patient Similarity Project**

Figure 1 shows the architecture of the project. The data used in the project comes from the Healthcare Graph Database at Optum. To obtain the data, a random walk query is performed over the graph. A number of random walks will be generated from the graph database. Each patient will be used as the source of a walk at least once to guarantee that every patient will be present in the input to the similarity algorithm.

Each walk from the random walk query will be a row of patients who are along some path. The source of the path will be the start of the row and the tail of the path is the end of the row as displayed in Figure 2.

A screenshot of a cell phone

Description automatically generated

**Figure 2. Output of a random walk**

The random walk paths received will be used as input to a Word2Vec model. Word2Vec is an embedding algorithm which normally takes words in a sentence and put them near each other in a low-dimensional space (typically a few hundred dimensions). In this case, patients along a path will be used as inputs to Word2Vec. This will put patients along the same path to be near each other in the embeddings.

The embeddings retrieved from the model can be re-inputted to the graph database. There is a FPGA mounted to the graph database that performs cosine similarity calculations on the data. The FPGA uses parallel computations to quickly find cosine similarity values between all patients in the database. In addition, the FPGA caches the data which allows for an even quicker computation time. The constant re-updating of new information in the FPGA cache and usage of the FPGA hardware quickly finds similarity between millions of patients that are embedded in low-dimensional space.

Tests were conducted by comparing values of cosine similarity between features and embeddings across 500 patients. The results are given below. Due to the high amounts of features, it’s expected that patients wouldn’t have very high cosine similarity in a dataset this small. The result’s key message is to note how similar the cosine similarity values are between the features and embeddings method. Additionally, Node2Vec performs better in terms of finding similar patients rather than dissimilar patients.

|  |  |  |  |
| --- | --- | --- | --- |
| **Patient 1 (EID)** | **Patient 2 (EID)** | **Cosine Similarity with Features** | **Cosine Similarity with Embeddings** |
| 56747958 (Breast Cancer Patient) | 56747958 (Same Patient for Sanity Check) | 1.000 | 1.000 |
| 56747958 (Breast Cancer Patient) | 7783729 (General  Healthy Patient) | 0.000 | 0.149 |
| 7792368 (General  Healthy Patient) | 7783729 (General  Healthy Patient) | 0.583 | 0.473 |
| 7528675 (Patient with Urinary Issues) | 7602763 (Patient with  Urinary Issues) | 0.302 | 0.342 |

**1.3 Details of Invention’s Novel Approach**

The invention is novel in its approach to storing and finding similarity between patients. This invention uses a graph database to store patient information instead of a table. This approach reduces time in querying patients who share features in the random walk query. Additionally, using a graph data base finds hidden relations between patients not found in tables.

Most similarity algorithms used in machine learning involve techniques such as K-Nearest Neighbors or cosine similarity, but these approaches are typically done in a table format. The usage of using graph similarity algorithms over table similarity algorithms is a unique property of this invention. The usage of a graph data base allows easy traversals between patients to find connections between them. This provided the inputs to the Word2Vec model that generates the embeddings. Although the concept of using traversals to get embeddings isn’t new, using this method in a healthcare setting is.

**1.4 Alternative methods of structuring the invention**

Although the main idea of the algorithm remains the same, the random walk process can yield different results on the same data if the graph structure is changed. When performing a random walk over a graph structure, it’s important to have a method of choosing the next patient. If patients are chosen through features they share, very rare features and very common features have an equal chance of being visited. To solve this, the random walk can be altered such that some features are more or less likely to be visited based on how common the feature is. This changes how likely a feature will be visited, which changes the structure of the random walk. The change in the structure of the random walk will change the outcome of the embeddings despite having the same data. Due to this, the invention can be altered based on the visiting likelihood.

**Additional Questions**

**2.1 Prior Submissions.** Is this related to any earlier UHG Invention Disclosure or any pending or registered UHG patent? If yes, please identify the disclosure(s) or patent(s).

No

**2.2 Public Disclosures.** Has the invention been used outside of UHG, have you disclosed the invention outside of UHG, or do you plan to do so? This would include using the invention in a pilot involving third parties, disclosing the invention to potential customers or suppliers, any publications, or any descriptions of any aspect of the invention to third parties (e.g., in a presentation, publication, abstract, advertisement, etc.). If so, please describe below. ***Note:*** *If currently unpublished, please keep the IP Legal Team updated on any future plans for non-confidential or public disclosures, publications or disseminations.*

No, this has not been used outside of UHG nor been disclosed to public. There are no plans to move this to public. The goal of this invention was to find whether patient similarity can be achieved through graph means.

**2.3 Current Commercial Use.** Is this invention currently implemented in a UHG product or solution? Y/N (circle one). If yes, please identify the product or solution: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Is the use external (used to provide products or services to others outside of UHG, or licensed to a third party (outside of UHG), or offered for sale, advertised, or marketed to a third party) or internal only? (Describe) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ No, it is not \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**2.4**  **Future Implementation**. If there is no current implementation of the invention, are there plans to implement it? If so, in what UHG product or solution is it likely to be implemented in? How do you anticipate that the invention will be used by UHG?

There is no current plans of implementing this invention.

**2.5 Competing Products / Solutions.** Please list the closest known competitive UHG or third party product(s) or solution(s).

N/A

**2.6 Third Party Contributions.** Was the invention developed in conjunction with any third parties (e.g., vendors, contractors or third party partners)? Does the invention include any third party materials, including software code? If so, please describe.

TigerGraph was used to store and retrieve information about the patients stored in the database. These Python libraries were used in the code I have wrote: pandas, pyTigerGraph, gensim, numpy, sklearn, matplotlib, scipy, streamlit, names. None of these were developed in conjunction with any third party libraries.

**2.7 Detection.** A patent is only enforceable if we can detect that another party is using the same process/system that we invented. How could UHG detect if a third party is using this invention? **Check all that apply and describe in detail how this would be accomplished.**

🞏 **Reverse engineering or code examination.** Based on the invention are we likely to have access to a third party’s code or other information that would allow reverse engineering? If so, how would we determine the entity is using this invention? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

🞏 **Testing a competitive solution/ view product operation.** Based on the invention what access are we likely to have to a competitive solution and how would we determine if that solution is infringing? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

🞏 **Reading product literature.** Based on the invention what type of third party product literature is likely to be produced and is it likely to provide detailed information that would identify whether a third party is infringing?

The implementation for the invention done is based on research. Finding literature on similar methods can be done. Due to its research nature, there is likely to be literature on patient similarity in general simply by searching for the keywords in Google.

🞏 **Impossible, unless 3rd party admitted use.**

🞏 **Don’t know.**

🞏 **Other:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Additions requested by the Patent Review Board (Oct. 1st, 2020)

Details on how this patent is novel.

We believe that the patent is novel due to the combination of various algorithms to solve a difficult problem: how to compare complex clinical data among 50 million patients in real time:

None of the following algorithms is unique in themselves:

1. Use of graph databases
2. Use of graph to store clinical data
3. Graph embeddings
4. Storing embeddings in graph
5. Caching embeddings in RAM
6. Random Walk
7. Use of FPGAs for parallel processing
8. Node2Vec
9. Conversion of healthcare events into features

When these components are combined, we get a unique solution.

In summary, we combined graph database technology, pre-computed embeddings stored in the graph at the member level, where there are multiple inventive steps needed to generate the embeddings:

* Random walk of the graph to determine members with similar conditions
* Algorithm to convert health care “events” into features – this includes code to summarize member’s conditions, medications, procedures, and claims, into features that can be treated like a feature vector.
* This includes some basic feature reduction, such as grouping medical conditions into their high-level groupings by walking up the ICD code hierarchy (e.g. Malignant neoplasm of right lung lower lobe is a more specific code then a general code for Lung Cancer)
* Once all of the embeddings are pre-computed, we can find similar members using FPGA in milliseconds from a huge member population

Other novel aspects being combined:

* We can increase or decrease the number of embedding to change performance, based on the requirements of speed and precision needed. The current value is 200 32-bit integers.
* At run time, decide to use a subset of the embeddings, if you only want to differentiate members into much more coarse-grained buckets, and speed is critically important we can reprogram the FPGA to only use the first 20 embeddings to subset the population.

# Importance of Speed for Patient Similarity

              Some of the algorithms described in this disclosure can be achieved using brute force methods of looping over millions of members and thousands of features and eventually getting to similar output in many hours, if not days or weeks.  So why is speed important, and why is it critical to have results in seconds?   In the healthcare domain, there will many scenarios in which speed of analytics is very important, and you don’t have luxury of knowing days in advance what you need to calculate.  For instance, you may have an Emergency Room setting, where a patient presents with a new set of diagnosed conditions, that weren’t previously in their medical record, and there would be no way to anticipate what took place.  So in this case, you want to know in real-time, what similar patients have experienced, how they were treated, and what outcomes they experienced.  Moreover, even if a patient comes in to see a doctor and the appointment was scheduled weeks in advance, they may share new symptoms with the doctor during that appointment, and the ability to run a new set of patient similarity analytics in real time is an important benefit of this approach.  In healthcare, speed is essential, and depending on outdated, obsolete data can have dire consequences.

              As we learned in the COVID crisis, treatments, outcomes, and patient comorbidities were changing on a daily basis, and this demonstrates the need to fast analytics, and insights derived from patient analytics in real time

The ideas in this disclosure are all very applicable in the healthcare setting.  Especially due to the vast amounts of information known about patients (medications, treatments, conditions, claims, demographics, etc.) and the critical nature of understanding how similar patients have been treated and progressed.  But the concepts in this disclosure could be applicable in any setting where there exists:

* A large, complex data model that describes an entity – this could be a user, a business, a legal document, a customer, etc
* The need to find similar entities, for the purpose of doing analysis of an entity and others like it
* The importance of speed and low latency results

For instance – an investment advisor may have a new customer walk in the door, and may want to use “customers like me” to recommend investment strategies, best stocks to own, average balances of savings, etc.  They could benefit from this approach for calculating similarity at massive scale and at high speed

# Application to Other Non-Clinical Domains

The problem if finding real-time similar items applies to many other fields. In many cases, the time-to-fix is critical to supporting hardware or software systems. In other cases, the user is using a web application and waiting for a response from a service where delays of multiple seconds will cause a user to have a negative experience.

## Log File Error Analysis

When an error occurs in a log file we want to know if that might be indicative of a potentially serious problem. The problem is often that a single ERROR, or WARNING message in a log file often only shows minimal information associated with a server or a device. We can use a graph of devices and their configuration information to enrich this error and then compare it to millions of similar errors on similar servers with similar version of software and containing similar mission critical information to accurately predict if this error warrants the attention of support staff. Note that the process of going from a device ID to context assumes that there is millisecond access to connected hardware data (an enterprise hardware graph).

A trivial example such as a WARNING message in a log file that of the disks is 95% full can allow an operator to remove old unused files before the system crashes resulting in a loss of service. The urgency of this fix may depend on the numbers of users impacted by services that depend on this resource. This urgency must be coded in the embeddings for this error to be correctly prioritized.

## Security Scanner Threat

This problem is similar to the prior problem about detecting an error in a log file with the key difference is that this is an ERROR or WARNING record generated by a security threat scanner. The reason this is different is because security threats have different types of codes and knowledge bases of known security threats that must be compared in real-time. The level of threats must also take into account complex system configuration information such as if a specific security patch has been installed or if another system configuration interacts with a specific threat.

## Helpdesk Problem Ticket Resolution

This problem is related to aiding a helpdesk quickly find solutions to problems in a knowledge base. For example, a user may provide a short description of a problem with their desktop application. This problem description can be encoded into a set of labels in a graph for the generation of the embedding and the compared to similar users with a similar desktop, OS version including security patches applied and other applications installed. All these factors will need to be encoded into a graph that represents the desktop configuration for calculating the embedding.

## Search Enablement

Many times, a user is looking for a specific document in a large knowledge base of documents. The user enters a search query specific keywords and they find a set of related documents, but it is many not exactly the specific document they are looking for if these keywords don’t appear in the document. Embeddings allow for the user interface to quickly show related documents even if the keywords were not present in the documents. Embeddings could allow knowledge search and retrieval systems to create distance calculations in higher dimensional knowledge spaces to find similar documents based on nearness to concepts, not exact keyword matching.