COVIDKG.ORG - a Web-scale COVID-19 Interactive, Trustworthy Knowledge Graph, Constructed and Interrogated for Bias using Deep-Learning

Bhimesh Kandibedala
Florida State University

Anna Pyayt
University of South Florida

Nickolas Piraino
Hemolix

Christopher Caballero, Michael Gubanov
Florida State University

ABSTRACT

We describe a Web-scale interactive *Knowledge Graph (KG)*, populated with trustworthy information from the latest published medical findings on COVID-19. Currently existing, socially maintained KGs, such as YAGO or DBPedia or more specialized medical ontologies, such as NCBI, Virus-, and COVID-19-related are getting stale very quickly, lack any latest COVID-19 medical findings - most importantly lack any scalable mechanism to keep them up to date.

Here we describe COVIDKG.ORG - an online, interactive, trustworthy COVID-19 Web-scale Knowledge Graph and several advanced search-engines. Its content is extracted and updated from the latest medical research. Because of that it does not suffer from any bias or misinformation, often dominating public information sources.

1 INTRODUCTION

Published medical knowledge doubles every 73 days, which makes prompt access to the latest, trustworthy, medical findings very challenging for both general public and medical professionals [36, 41–44]. This problem is especially noticeable during the pandemic, when all popular, previously trusted sources - the Web, Social Media, News Media suddenly become full of biased opinions and misinformation. Such remarkable information "decay" is fueled by both the severity of the problem and desperate need for trustworthy information in order to survive. At the same time, the current lack of a viable technology for collecting and conveniently accessing all up to date trusted medical knowledge, results in time-consuming
Google/PubMed/QxMD/other search, aggravated by the need to read hundreds of returned Web-pages, publications, which is prohibitively slow and usually still does not have the up to date information indexed [16, 32, 34, 38, 39, 66, 75, 77]. The innovation is a technology that can be used to automatically construct and refresh a Scientific Knowledge Graph (KG) having all latest trustworthy, vetted medical knowledge. Having had it operational before and during the COVID-19 pandemic could have been game changing for our society. It can help survive during any future pandemics as well - be it Monkeypox, Polio, Zika, Ebola, or a new unknown virus.

Currently existing, socially maintained KGs such as YAGO [70], DBPedia [19] or medical ontologies, such as NCBI, Viral [8], COVID-19 [2] and other are static, hence all are quickly becoming stale and lose their value. Manually curated popular resources such as CDC.gov and WebMD.com are updated more frequently, but are very high-level and can afford to cover only the most dominating topics due to high update cost. Resources such as covidgraph.org [10] focus on a set of very narrow topics in COVID-19 genetics, merged with older SwissProt, Gene Ontology ontologies. They have slightly deeper knowledge, but limited to a very narrow subtopics. By contrast, we are proposing a new Deep-learning (DL) architecture, capable of constructing and refreshing our Web-scale KG on demand for a given domain that will exhibit both broad topical coverage within the domain, as well as depth within each topic. We take COVID-19 as a model, but without making our architecture depend on it, so the overall approach remains truly "on demand" – i.e. applicable to other scientific areas.

Our COVIDKG.ORG system bridges the gap between the current shortcoming solutions of trustworthiness and ease of comprehension [17, 54, 55, 57–59, 78]. It does so by coupling the user-friendly KG interface with the actively maintained and interrogated for bias training datasets, full of new vetted, medical research findings [79].

This paper describes the technology and our experience with COVIDKG.ORG Web-scale KG. It is on the World Wide Web with several advanced working Search Engines [13, 20, 35, 45–47, 51, 56, 69, 74]. One of the sources is CORD-19 data set [79] represents-one of the most extensive machine-readable coronavirus research literature collection available.

We will begin by describing some of our background knowledge work and research that we have learned about the topic from our own experience and the endeavors of other public health informatics organizations. Then extensively detail the specifications of our back-end and front-end data systems and their communication. We will then describe some of our Machine- and Deep-Learning models and the process we designed to train them at scale. These models analyze and discover new findings from our datasets, extract new COVID-19 findings and enrich the graph. We finish by extensively reviewing some of the similar systems currently available in the world [5, 14, 15, 21, 22, 25–28, 31, 33, 36–38, 40, 41, 48, 49, 54, 60, 62, 67, 68, 71].

2 ARCHITECTURE

After researching the current COVID-19 projects and Knowledge Graphs [2, 19, 24, 29, 70, 76, 80] as well as investigating public needs through conducting hundreds of interviews and customer discovery process [11], we have designed, validated, and launched
COVIDKG.ORG. The architecture is demonstrated in Figure 1. №1 in the Figure represents a Medical Engineering professional who creates an initial, small (10-20 nodes) structural layout that will initialize the base of our Knowledge Graph. №2 corresponds

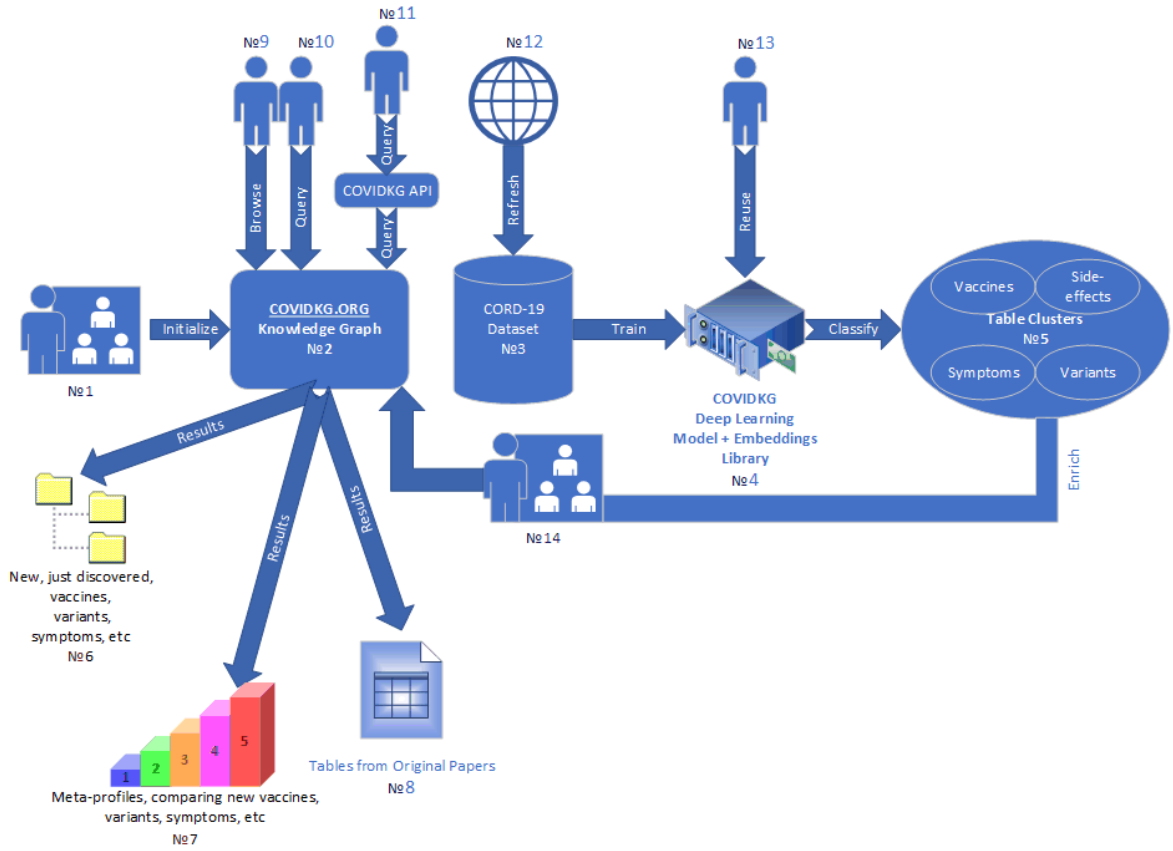

Figure 1: COVIDKG.ORG Architecture.

to our current Knowledge Graph, stored in a scalable sharded MongoDB storage [9]. №3 depicts the CORD-19 dataset [79], parsed, processed by our trained Deep-learning models, and also stored in a sharded MongoDB. №4 represents a high performance NVidia GPU cluster, responsible for training and classification workloads of our Deep-Learning models and custom tabular embeddings. It is configured with Apache Spark MLLib [18] and TensorFlow [12]. №5 on the Figure shows the topical clusters that are categorized from the dataset by relevant COVID-19 topics. №6 corresponds to newly discovered vaccines, strains, side-effects extracted by our Deep-Learning models from the dataset later fused with the main KG. №7 corresponds to our Deep learning models, learning the multi-layered 3D Meta-profiles, summarizing and visualizing knowledge from several sources. E.g. Figure 6 displays a multi-layered 3D profile for *COVID-19 Vaccine Side-effects* composed from three different COVID-19 papers. This 3D visualization summarizes information from 9 different sources in one place and is much easier to comprehend than reading these 3 papers and understanding all details about the vaccine side-effects. №8 corresponds to the original tables. №9, 10 represent users who browse the Knowledge Graph by clicking nodes and using the interactive features or query our custom search-engines. №11, 13 are the COVIDKG API users that might want to query the Knowledge Graph or fine-tune and reuse our released, pre-trained Deep-learning models or Embeddings on their own dataset. №12 depicts the World Wide Web with new information on COVID-19. №14 portrays the fusion of sub-trees having several layers or addition of new nodes that may have to be evaluated by a human expert before the fusion into the Knowledge Graph.

COVIDKG.ORG back-end consists of a sharded MongoDB JSON storage [9] that holds more than 450,000 publications on COVID-19 from CORD-19 [79] as well as other sources, parsed into JSON and enriched with different classified characteristics by our Deep-Learning models, running non-stop, classifying new incoming publications.

The user-facing component of the COVIDKG.ORG system is the interactive Knowledge Graph front-end graphical Web interface. It displays and allows convenient interaction with the hierarchical structure of nodes and edges that display valuable information learned from the CORD-19 dataset as well as vetted information from other reputable COVID-19 API resources. COVIDKG.ORG also hosts several custom search-engines to process complex queries and retrieve more detailed information from the original publications. COVIDKG.ORG also releases hundreds of pre-trained models and embeddings as an API for reuse by data scientists and developers.

Datasets: One of the datasets that our system utilizes is CORD-19 - an open research dataset [79], which adheres to high academic peer-review trustworthiness and relevance standards that public health informatics require. COVID-19 Research Dataset (CORD-19), was put together by the Office of Science and Technology Policy in the White House, along with six other institutions. Their goal was to revolutionize modern medicine by encouraging researchers with proper access to a vetted dataset

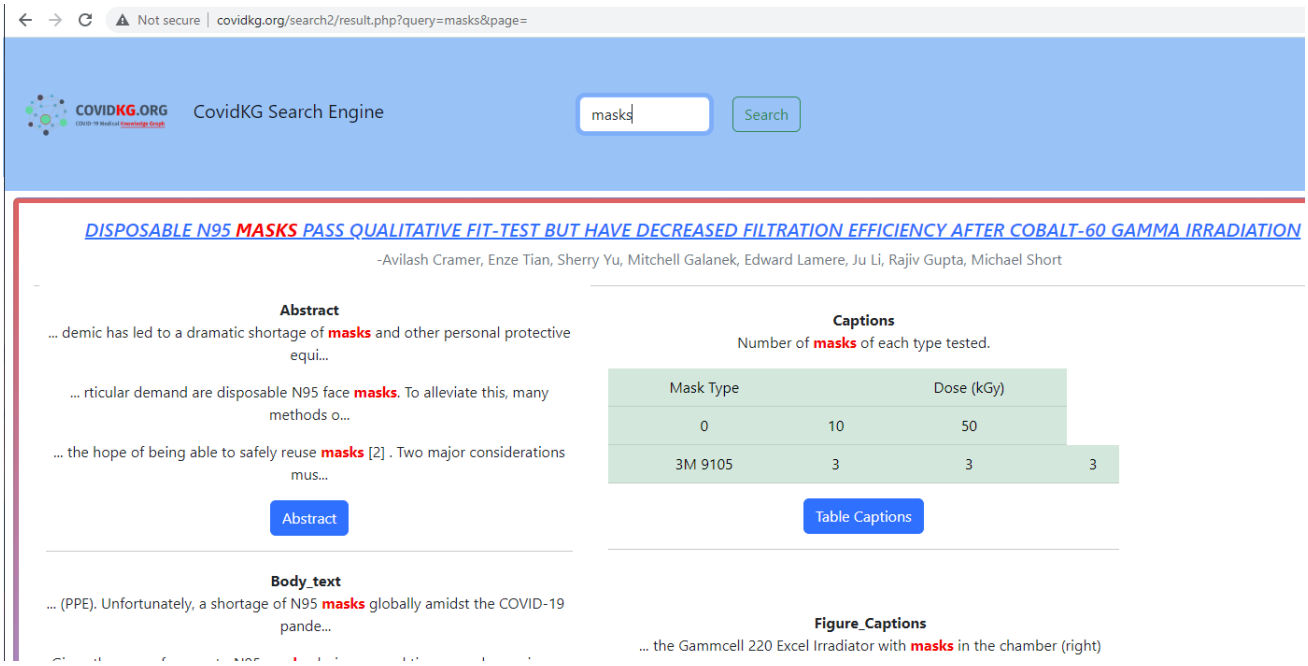

Figure 2: COVIDKG.ORG Advanced Publication Search-engine Interface.

to develop question-answering systems for progression in significant COVID-19 discoveries. Enabling the general public access to text and data mining on research articles without violating the rights of the authors. During its first few months more than 3,500 new publications were updated per week. This dataset has had continuous growth with new publications from the top major publishers in medicine and has grown to over 450,000 open-access publications [79].

Storage: Our MongoDB [9] sharded cluster storing data and all trained Deep-learning models and embeddings takes ≈965GB for its distributed dataset storage, with raw space consumption of more than 5TB.

2.1 Advanced Search-engines

CovidKG.ORG hosts several advanced search engines. The purpose of these search engines is to enable anyone from scientists to researchers and even to general public users, to have access to the latest cutting edge information on COVID-19 and related scientific research. We currently provide three different search engines for different types of structural queries. All three have a similar evaluation process, but produce different sets of results. Each one allows for exact match of the query if wrapped in quotes or stemming match capability on a tokenized query. The Search Engine receives results from the database by using an aggregation query that passes the data through a series of pipeline stages. The first stage in the pipeline is a "$match" expression. It allows the developer to specify a condition to filter the dataset to pass on to the next stage in the pipeline. Our $match pipeline stage utilizes text-based search through regular expressions that are stemmed from the root users searched terms. It was mindful to use the $match stage first to minimize the amount of data being passed through all the latter stages, thus significantly increasing performance and response time to the user. In the next stage, the data is passed through a $project stage, which streams only the specified fields to the next stage of the aggregation pipeline. So we only specified fields that were necessary for carrying out calculations and printing to the screen. By removing unnecessary fields that take up space and time passing through each proceeding stage we significantly improve our systems performance. The pipeline also uses a few custom "$function" stages to derive calculations based on the individual documents and the searched query for ranking results. These custom functions are written in JavaScript inside of MongoDB aggregation pipeline query. Once the aggregation is finished the results are paginated as a list of ten per page displaying brief snippets of the document and access to the full text. The ranking is an accumulation of various weighted features per document, such as the number of matches, proximity between the matched terms and which field the term was matched in. Each term in the corpus has an associated Term Frequency-Inverse Document Frequency (TF-IDF) [53] weight in order to reward more important terms. For each matched term its TF-IDF is weighted in the ranking per document.

2.1.1 Search over Paper Title, Abstract, Caption. This search engine has three search fields for title, abstract and table captions. The search fields are inclusive in the search results, meaning, if a user searches on a field there must be a document that matches at least one term in that field or it does not get passed on to the next stage regardless if there are matches over the other fields. The results are formatted with table captions first, the title and authors and the full abstract.

2.1.2 Search over all Publication Fields. If the user is unsure of where exactly the term may be or where the term is referenced is unimportant to the results then search over all fields is a good fit. As shown in Figure 2, which depicts a screenshot of results where a user queried for "masks". These results are formatted with a brief excerpt of where it matches in abstract, body text, table captions, tables and figure captions. The interface also allows the user to expand and collapse appropriately.

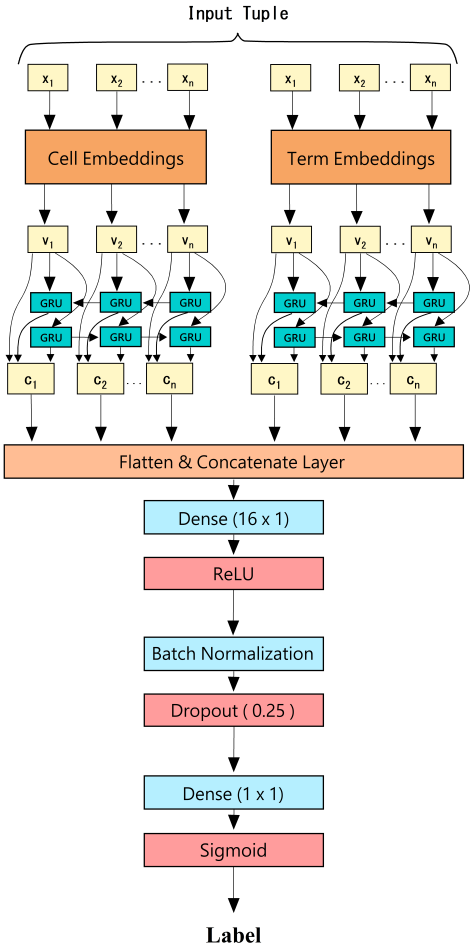

Figure 3: Deep-learning BiGRU Architecture with Paralell Term- and Cell-level Embedding Layers.

2.1.3 Search over Paper Tables. Search capability over tables is a large part of our intentions to enable advanced structural information retrieval. These search results are a product of regular expression search over table captions and all of the table's data and displayed, as shown in Figure 4. The figure depicts a screenshot of search results from a user query "ventilators" and displays results where ventilators matched the tables. The display highlights, in red, the matched term for every field. As shown, there is a match at the first table and the abstract. The screenshot is cut off at the bottom due to space constraints, but there are more search results, which are ranked with an advanced ranking function having both static and dynamic features. Each field displays a brief view of the matches and allows the user to expand and collapse respectively.

3 METADATA CLASSIFICATION

Hardware: Training and validating of some of our models were done on a cluster of 4 machines, each having 4 Intel Xeon 2.4Ghz 40-core CPUs, from 192GB to 1TB of RAM, with 10TB disk space each, interconnected with a 1GB Ethernet. All software was written in the Python programming language. For implementing the RNN model, we have used Keras, with Tensorflow framework as the backend. The SVM classifier was implemented using Scikit-learn, a popular machine learning library for python.

3.1 CORD-19

The COVID-19 Open Research Dataset (CORD-19)[79] is an extraction of scientific papers on COVID-19 [79]. In addition to the paper's fields (i.e. authors, title, abstract), it also contains raw metadata. We developed an additional HTML table parser and post-processor that takes raw HTML fragments from CORD-19 and converts them to semi-structured, clean JSON[66] format.

3.2 Feature space

We have used 100'000 dimensional feature space, i.e. 100K English terms in our vocabulary that we have selected by taking all terms from our datasets, sorting by frequency and cutting off the noise words and spam [78]. Increasing the dimensionality further led to significantly slower training time, which would prevent or make the experiments much more difficult.

3.3 Evaluation

We composed the training sets from Web-scale datasets such as WDC [61] and CORD-19 respectively [79]. We evaluated our models and observed 89% - 96% F-measure on average respectively, when validated with 10-fold cross-validation, for Machine-learning-based model (SVM [63]) and Deep-learning Bi-GRU-based models with slight differences depending on whether the classified metadata is horizontal or vertical, as well as its row/-column number.

3.4 Pre-processing

To streamline the processing of numerical data handled by the model, we have created several regular expressions that encode all numerical data falling in similar forms under its relevant category. The substitution is done as follows.

All the *Zeros* in both decimal and integer forms in the data are substituted with `ZERO`. The order of these expressions is important as 0 in 50 is not the same as 0.0. The data in the form of arithmetic ranges like *5-10 mg*, is replaced with the keyword `RANGE`, However, we have not replaced the units following the range as they are tackled in the later part of the script. The magnitude of data in the dataset is not uniformly spread. A large part of data is the numbers valued less than 0, so we have divided these numbers of different magnitudes into three parts. The negative integers are replaced with `NEG`, this expression only takes negative numbers and not the words/ranges with - in them. The numbers less than 0 are replaced with `SMALLPOS`. The numbers greater than 0 are further divided into two parts, `FLOAT` and `INT`, these numbers have no limit and are not further binned as we didn't observe any pattern with upper limits. Now, after having substituted all numbers, we are left with symbols, units, and operators. We have replaced % with `PERCENT`. Note that 5% and 0.5% will not be replaced the same way, Their respective substitutions will be `SMALLINT PERCENT` and `INT PERCENT`. The dates of the form where month is represented in words are substituted with `DATE`, However, we are not handling the dates of the form *mm/d-d/yy*. Symbols < and > are substituted with `LESS` and `GREATER` keywords respectively. The most frequently occurring units were *Time, ml, mg, and kg* so we have substituted the integers followed by these units in their in their respective descriptive keywords.

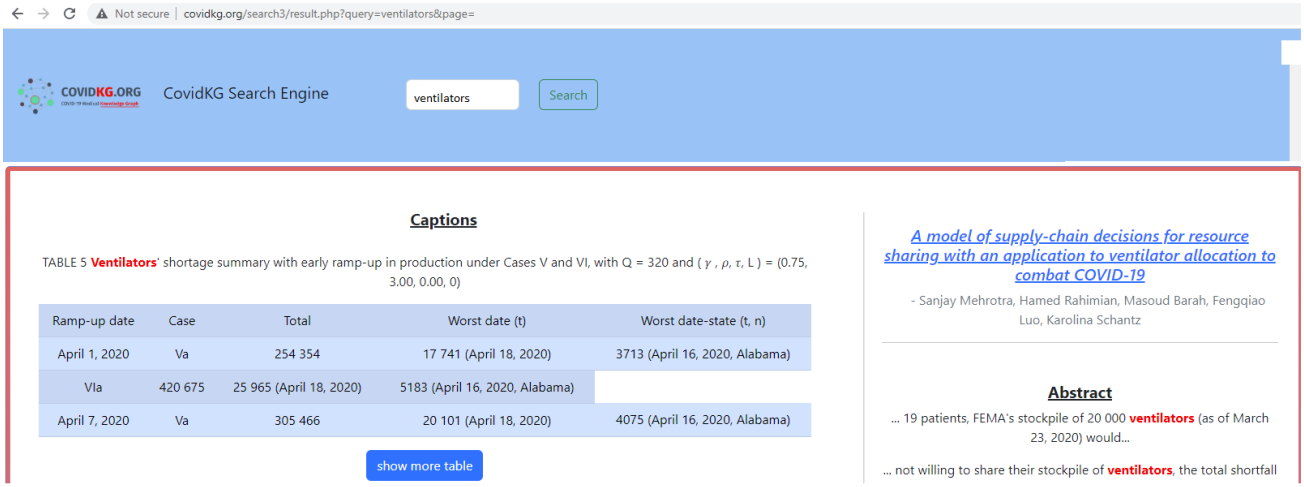

Figure 4: COVIDKG.ORG Advanced Table Search-engine Interface.

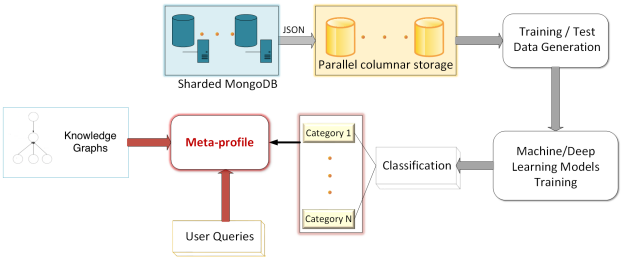

Figure 5: Back-end Architecture.

3.5 New Positional Features

We construct the feature vector by calculating new positional features from each row of the table. We use these feature vectors for the SVM model [63]. The feature vector consists of 7 features $\{f_1, f_2, ..., f_7\}$ where f_1 is a data or metadata row with valid numerical substitutions as described in the pre-processing step, f_2 is the number of cells in the table row, f_3 is a binary value conforming if the above row exists for the current row, f_4 is a binary value conforming the above row below exists for the current row, f_5 equals to the total number of cells in the row above, f_6 equals to the total number of cells in the below row , f_7 is a boolean label indicating if it is a metadata row (NULL for the training instances). $\{f_3, .., f_7\}$ all the features collectively are called *positional* features. Each feature affect the metadata classification outcome.

3.6 BiGRU Ensemble with parallel Embedding Layers

Figure 3 depicts the architecture of a BiGRU ensemble consisting of three main stages. In the first stage, a data or metadata tuple, $\{x_1, x_2, ..., x_n\}$, where x_i is the i^{th} term from the tuple, is pre-processed to create both cell-wise and term-wise representations. It includes data cleaning along with the replacement of numbers and ranges in data with placeholders such as *NUM, RANGE*, etc as is described above for our Machine-learning model. The pre-processed feature vectors are then used to to train Word2Vec embeddings [65] on the whole corpus (we pre-trained on WDC and CORD-19 and then fine-tuned with end-to-end training on the target corpus). The model runs along both inputs in parallel, converting them into their respective pre-trained embedding vectors, $\{v_1, v_2, ..., v_n\}$. This sequence is passed through a BiGRU layer with 100 BiGRU units and the result is concatenated with the original embeddings to create our enriched contextualized vectors, $\{c_1, c_2, ..., c_n\}$. We flatten the output of each path to create both cell- and term-wise tuple representations. The final stage of the model concatenates the two representations and passes them through a dense layer of 16 units, a batch normalization layer, a dropout layer and a dense binary classifier.

Since tuples in tables are order independent and context specific, both global average pooling and traditional RNNs are ill-suited for creating good tuple representations. Because of this we tried bidirectional RNNs (biLSTM and biGRU), since they have been shown to capture contextual dependencies by taking into account both forward and backward context [50] [72]. This not only reduces the effect of order dependence inherent to traditional RNNs but also captures the context specific information that is lost in averaging over the static word embeddings, [52]. We opted for the biGRU layers over biLSTM because while performance was slightly worse, with -0.02 ΔF1-Score, -0.07 ΔPrecision, +0.06 ΔRecall, the training time was faster. Concatenating the original embeddings with their context specific representations allows the model to additionally account for global correlation when making predictions.

4 KNOWLEDGE GRAPH

4.1 Initialization

The structural hierarchy (i.e. nodes and edges) for the Knowledge Graph will be initialized with the help of a Medical expert (№1 in Figure 1). On the highest level, the general characteristics of COVID-19 as a virus can be extracted from older, vetted ontologies about viral infections, e.g. symptoms, ways of transmission, etc. Once initialized, the KG will get automatically updated from the vetted medical sources. This ensures reliability, freshness, and quality of our KG (i.e. №2 in Figure 1).

4.2 Enrichment and Fusion

Once the KG initialization is complete we fuse the extracted information into our Knowledge Graph during the enrichment process. We classify and extract the clusters prominent COVID-19 topics (e.g. №5 in Figure 1). This process is challenging since

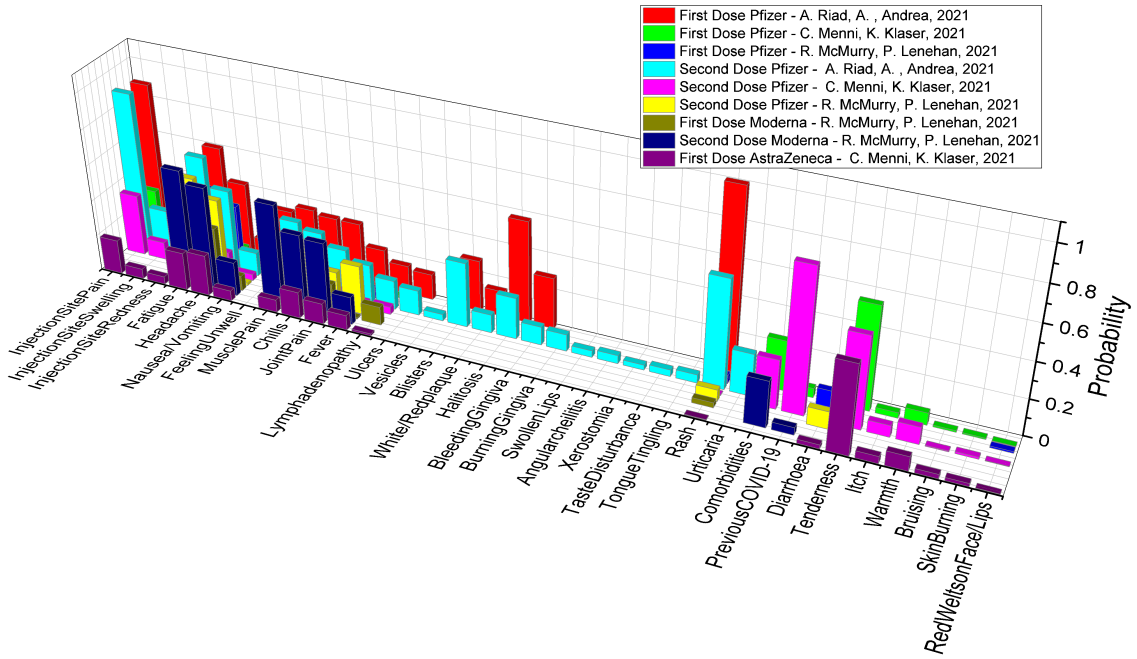

Figure 6: Meta-Profiles for COVID-19 vaccination side-effects, extracted from tables in three papers, grouped by vaccine, dosage, and paper [23, 64, 73].

all topical clusters have different structure and significant concepts and terms can be referred to differently (e.g. *COVID-19* and *coronavirus disease 2019*). Consequently, we trained a variety of advanced AI models with our new tabular embeddings to help perform accurate clustering [30, 57].

The graph is populated with nodes and edges and is stored in JSON format. The structure of the graph is hierarchical, so all child nodes have parent nodes. The user can search over the KG via the front-end interface that except matching nodes also highlights the path to the matching nodes. The user can then either browse the graph to explore the structure starting from the matching nodes or click the papers linked off these nodes to read about the topic of preference in more detail.

Fusion of the extracted hierarchical knowledge into a segment or several segments of our KG requires taking into consideration multiple levels of abstraction. For example, "Symptoms" can be a node in a subtree "Clinical presentation" that could be, in turn, linked to the "COVID-19" KG root node. Because of the different ways to categorize, the actual symptoms may overlap in different KG subtrees. After consulting with several medical experts it was decided to store all different ways to categorize the data without merging them, since each of the categorization methods can be useful for different kinds of users and medical specialists. While general public might be interested in common and rare symptoms, medical specialists might analyze specific organ systems. For example, sorting by "rare symptoms" and "common symptoms" can overlap with the sets of symptoms sorted by "organ systems". In addition to that, even though "Neurological symptoms" are related to the nervous system in general, while "Cerebrovascular" is related to the brain and its blood vessels, they have significant overlap in symptoms. The first step of fusing the extracted hierarchical knowledge into the KG is matching the root node of the extracted subtree to the corresponding node(s) in the KG. This matching process is based on normalized NLP term matching, amended by the embedding-driven matching. The latter is especially important in context of new terms, unseen before, which is often the case with new vaccines, viral strands, etc. For example, assume we have extracted a subtree Vaccine → NovoVac from the table's metadata. The root node Vaccine may match to the KG node Vaccine(s) by normalized NLP term matching and then the leaves (NovoVac) can be merged with the leaves of the matched node in the KG. However, if there is no corresponding KG node Vaccine(s) and there is no match to the KG leaves with existing vaccines, the embedding vector corresponding to the new vaccine (NovoVac) extracted from metadata can be used to match it to the embeddings vectors of the existing vaccines in the KG due to them being close to each other by distance. The node Vaccine then can be added to the KG on the top of the NovoVac node. If the extracted subtree has several layers of hierarchy, e.g. Side-effects → Children side-effects → Rash, Rash has to be left separate from the existing side-effects in the KG, even if matched to them by having close embedding vectors. This is because, it is categorized as Children's side-effects, which is a separate category from regular side-effects, so both the new node Children's side-effects and its leaves have to be added to the KG, even if some of the side-effects overlap with the general side-effects,

already present in the KG. Fusion of sub-trees, having several layers or insertion of new nodes will have to be evaluated by a human expert (№14 in Figure 1); fusion of leaves with nodes matched with high confidence score may be left unsupervised. Over time, all categories of initial fusion mistakes identified by the expert will be learned by the fusion module to be automatically corrected, hence most of the fusion is expected to become minimally supervised.

5 RELATED WORK

[6] is an Information Retrieval (IR) system over publications at researchrabbit.ai. They are introducing a retrieval mechanism over papers that does not require the use of keyword-search. They display a force directed graph of related, cited and referenced papers that a user can construct and use. They provide many features, such as being able to create your own custom graph of papers, curated collections to improve recommendations, personalized alerts, sharing and collaborating of papers and graphs, and among others the ability to discover author networks. Finally, ResearchRabbit is a free service.

Another system authored by the Center for Artificial Intelligent Research, HKUST ([1]), available at demo.caire.ust.hk/covid/. CAiRE-COVID is also a free service and aims to provide a resource for solutions to the *coronavirus* disease by using a Machine-learning based system that utilizes NLP question answering techniques along with the summarization to help discover available scientific literature. Their system is comprised of 3 modules. The pipeline begins with a user query sent through the first module - document retrieval, which does paraphrasing and search. Query paraphrasing converts the user query to several shorter and simpler analogous questions. The search engine takes advantage of Anserini and Lucene to retrieve related publications. Then the snippet selector module finds the related evidence among the whole text by using answer re-ranking, and term matching score. Finally a "Multi-document abstractive summarizer" that synthesizes the answer from multiple retrieved snippets steps in and generates the final answer.

Another relevant system - [7] is available at http://covidsearch.sinequa.com. Sinequa has access to a COVID-19 Intelligent Insight portal of over 100,000 curated scientific publications. Sinequa's search engine supports full-text search using NLP. The Search engine supports ranking by relevance and recognizes synonymy in their ranking function. It has 3 sections. One for the matched scientific results, one for showing more details on a selected result and the last one for filtering and sorting the result set. The system highlights important information throughout each result and tags them all by different classification labels. Sinequa's system is also provided for free.

[3] at http://covidscholar.org is an information retrieval resource for Covid-19 and related scientific research, established by Matscholar's research effort. COVIDScholar also uses NLP to empower search over a COVID-19 related dataset. The search results are matched by title, abstract and keywords. COVIDScholar displays title authors and abstract, while providing a link to the full-text at its original publisher and a list of related works. They neither have a KG, nor an advanced search unlike us.

[4] at [10] provide a free and public GraphQL API. Their KG is populated from certain classic, well-known ontologies such as NCBI, UniProt and other sources. Their graph has information on genes of interest, transcripts, protein identifiers function names and gene names from many different resources.

CovidKG.ORG provides several advanced search-engines over COVID-19 scientific resources. The user can either search the KG, or over all sections of the original publication, just the title, abstract or table captions or just the table data. The search-results page provides a list of ranked scientific resources with access to each full-text of each section of the paper, full-text of the whole document, and ranked tables with the most relevant results. The ranking function incorporates matching terms and synonyms, proximity, document, terms, and publication weights, as well as many others. CovidKG.ORG classifies the documents by related topics enabling the data to be further categorized. The advanced search-engine over tables displays a brief section with the most relevant tables first that can be expanded to see more results. The CovidKG.ORG Knowledge Graph is a complex interactive hierarchical data structure fused from all relevant research results found in the rich corpus of scientific resources that we curate. The KG is trustworthy as it is built only from the vetted knowledge. It supports interactive search through paths of nodes that allows getting complex insights into the *provenance* of the search result. The nodes along the path provide access to the publications, where the result is coming from.

6 ACKNOWLEDGMENTS

We would like to thank anonymous reviewers for their valuable feedback, FSU, NSF (Award №2229256), and Amazon for their support of this research project.

7 CONCLUSION

Here we described COVIDKG.ORG - the first, interactive, trustworthy, Web-scale Knowledge Graph on COVID-19 Medical Knowledge. We highlighted here its front- and back-end architectures, the Artificial Intelligence models behind it that are constructing and keeping it up to date non-stop. It is extracted from vetted, latest medical research sources, hence does not suffer from any bias or misinformation unlike many public information sources.

REFERENCES

[1] [n.d.]. CAiRE-COVID.
[2] [n.d.]. The COVID-19 Infectious Disease Ontology. https://www.ebi.ac.uk/ols/ontologies/idocovid19.
[3] [n.d.]. COVIDScholar. https://covidscholar.org/stats
[4] [n.d.]. HealthECCO. https://healthecco.org/covidgraph/
[5] [n.d.]. online: http://www.recordedfuture.com. http://www.recordedfuture.com
[6] [n.d.]. researchrabbit. https://www.researchrabbit.ai/.
[7] [n.d.]. Sinequa. https://covidsearch.sinequa.com/app/covid-search/#/home
[8] [n.d.]. The Virus Infectious Disease Ontology. https://www.ebi.ac.uk/ols/ontologies/vido.
[9] 2007. online: http://www.mongodb.com. http://www.mongodb.com
[10] 2022. online:Helathecco medical graph. Healthecco.org/covidgraph/
[11] 2022. online:The National Science Foundation's Innovation Corps (I-Corps™) program. https://www.nsf.gov/news/special_reports/i-corps/
[12] Martín Abadi. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. http://tensorflow.org/ Software available from tensorflow.org.
[13] Zia Abedjan, John Morcos, Michael Gubanov, Ihab F. Ilyas, Michael Stonebraker, Paolo Papotti, and Mourad Ouzzani. 2014. DATAXFORMER: Leveraging the Web for Semantic Transformations. In *CIDR*.
[14] Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting Relations from Large Plain-Text Collections. In *ACM DL*. citeseer.ist.psu.edu/agichtein00snowball.html
[15] E. Agichtein, P. Ipeirotis, and L. Gravano. 2003. Modeling query-based access to text databases. citeseer.ist.psu.edu/agichtein03modeling.html
[16] Bogdan Alexe, Michael Gubanov, Mauricio A. Hernández, C. T. Howard Ho, Jen-Wei Huang, Yannis Katsis, Lucian Popa, Barna Saha, and Ioana Stanoi. 2008. Simplifying Information Integration: Object-Based Flow-of-Mappings Framework for Integration. In *BIRTE*.
[17] Bogdan Alexe, Michael Gubanov, Mauricio A. Hernandez, Howard Ho, Jen-Wei Huang, Yannis Katsis, and Lucian Popa. 2009. Simplifying Information

Integration: Object-Based Flow-of-Mappings Framework for Integration. In *Business Intelligence for the Real Time Enterprise*. Springer.
[18] Michael Armbrust, Reynold S. Xin, Cheng Lian, Yin Huai, Davies Liu, Joseph K. Bradley, Xiangrui Meng, Tomer Kaftan, Michael J. Franklin, Ali Ghodsi, and Matei Zaharia. 2015. Spark SQL: Relational Data Processing in Spark. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (Melbourne, Victoria, Australia) *(SIGMOD '15)*. ACM, New York, NY, USA, 1383–1394. https://doi.org/10.1145/2723372.2742797
[19] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *ISWC'07/ASWC'07*.
[20] Zohra Bellahsene, Angela Bonifati, and Erhard Rahm. 2011. Schema Matching and Mapping. In *Springer*.
[21] Michael J Cafarella, Alon Halevy, Daisy Zhe Wang, Eugene Wu, and Yang Zhang. 2008. Webtables: exploring the power of tables on the web. *VLDB* (2008).
[22] Michael J Cafarella, Dan Suciu, and Oren Etzioni. 2007. Navigating Extracted Data with Schema Discovery.. In *WebDB*. Citeseer.
[23] Anna May Lorenzo Polidori Joan Capdevila Panayiotis Louca et al Cristina Menni, Kerstin Klaser. 2021. Vaccine side-effects and SARS-CoV-2 infection after vaccination in users of the COVID Symptom Study app in the UK: a prospective observational study.
[24] Google Developers. 2012. Google Knowledge Graph. https://developers.google.com/knowledge-graph.
[25] Jörg Diederich, Wolf-Tilo Balke, and Uwe Thaden. 2007. Demonstrating the semantic growbag: automatically creating topic facets for faceteddblp. In *JCDL*.
[26] AnHai Doan, Pradap Konda, Paul Suganthan G. C., Yash Govind, Derek Paulsen, Kaushik Chandrasekhar, Philip Martinkus, and Matthew Christie. 2020. Magellan: toward building ecosystems of entity matching solutions. *CACM* (2020).
[27] Xin Luna Dong, Barna Saha, and Divesh Srivastava. 2013. Explaining data fusion decisions. In *WWW*.
[28] D. Downey, O. Etzioni, S. Soderland, and D.S. Weld. 2004. Learning text patterns for Web information extraction and assessment. In *AAAI*. citeseer.ist.psu.edu/agichtein03modeling.html
[29] Christiane Fellbaum (Ed.). 1998. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press.
[30] Anna Lisa Gentile, Petar Ristoski, Steffen Eckel, Dominique Ritze, and Heiko Paulheim. 2017. Entity Matching on Web Tables: a Table Embeddings approach for Blocking. In *EDBT*.
[31] Yash Govind, Pradap Konda2, Paul Suganthan G. C., Palaniappan Nagarajan, Han Li, Aravind Soundararajan, Sidharth Mudgal, Jeffrey R. Ballard, Haojun Zhang, Adel Ardalan, Sanjib Das, Derek Paulsen, Amanpreet Saini, Erik Paulson, Youngchoon Park, Marshall Carter, Mingju Sun, Glenn M. Fung, and AnHai Doan. 2019. Entity Matching Meets Data Science: A Progress Report from the Magellan Project. In *SIGMOD*.
[32] Michael Gubanov. 2017. Hybrid: A Large-scale In-memory Image Analytics System. In *CIDR*.
[33] Michael Gubanov. 2017. Polyfuse: A large-scale hybrid data fusion system. In *ICDE*.
[34] M. Gubanov, C. Jermaine, Z. Gao, and S. Luo. 2016. Hybrid: A Large-scale Linear-relational Database Management System. In *MIT NEDB*.
[35] Michael Gubanov, Chris Jermaine, Zekai Gao, and Shangyu Luo. 2016. Hybrid: A Large-scale Linear-relational Database Management System. In *MIT Annual DB Conference*.
[36] Michael Gubanov, Manju Priya, and Maksim Podkorytov. 2017. CognitiveDB: An Intelligent Navigator for Large-scale Dark Structured Data. In *WWW*.
[37] Michael Gubanov and Anna Pyayt. 2012. MedReadFast: Structural Information Retrieval Engine for Big Clinical Text. In *IRI*.
[38] M. Gubanov and A. Pyayt. 2013. ReadFast: High-relevance Search-engine for Big Text. In *ACM CIKM*.
[39] M. Gubanov and A. Pyayt. 2014. Type-aware Web search. In *EDBT*.
[40] Michael Gubanov, Anna Pyayt, and Sophie Pavia. 2022. Visualizing and Querying Large-scale Structured Datasets by Learning Multi-layered 3D Meta-Profiles. In *BigData*. IEEE.
[41] M. Gubanov, A. Pyayt, and L. Shapiro. 2011. ReadFast: Browsing large documents through UFO. In *IRI*.
[42] Michael Gubanov and Linda Shapiro. 2012. Using Unified Famous Objects (UFO) to Automate Alzheimer's Disease Diagnostics. In *BIBM*.
[43] Michael Gubanov, Linda Shapiro, and Anna Pyayt. 2011. Learning Unified Famous Objects (UFO) to Bootstrap Information Integration. In *IRI*.
[44] M. Gubanov and M. Stonebraker. 2014. Large-scale Semantic Profile Extraction. In *EDBT*.
[45] M. Gubanov and M. Stonebraker. 2014. Text and Structured Data Fusion in Data Tamer at Scale. In *ICDE*.
[46] Michael N. Gubanov and Philip A. Bernstein. 2006. Structural text search and comparison using automatically extracted schema.. In *WebDB*.
[47] Michael N. Gubanov, Philip A. Bernstein, and Alexander Moshchuk. 2008. Model Management Engine for Data Integration with Reverse-Engineering Support. In *ICDE*.
[48] A. Halevy. 2013. Data Publishing and Sharing using Fusion Tables. In *CIDR*.
[49] Joseph M. Hellerstein, Christopher Re, Florian Schoppmann, Daisy Zhe Wang, and Eugene Fratkin. 2012. RuleMiner: Data quality rules discovery.. In *PVLDB*.
[50] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. arXiv. https://doi.org/10.48550/ARXIV.1508.01991
[51] Kazi Islam and Michael Gubanov. 2021. Scalable Tabular Metadata Location and Classification in Large-scale Structured Datasets. In *DEXA*.
[52] L. C. Jain and L. R. Medsker. 1999. *Recurrent Neural Networks: Design and Applications* (1st ed.). CRC Press, Inc., USA.
[53] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* 28 (1972), 11–21.
[54] Rituparna Khan and Michael Gubanov. 2018. Nested Dolls: Towards Unsupervised Clustering of Web Tables. In *IEEE Big Data*.
[55] Rituparna Khan and Michael Gubanov. 2018. Towards Unsupervised Web Tables Clustering.. In *IEEE BigData*.
[56] Rituparna Khan and Michael Gubanov. 2020. Towards Tabular Embeddings, Training the Relational Models. In *IEEE Big Data*.
[57] Rituparna Khan and Michael Gubanov. 2020. WebLens: Towards Interactive Large-Scale Structured Data Profiling. In *CIKM*.
[58] Rituparna Khan and Michael Gubanov. 2020. WebLens: Towards Interactive Web-scale Data Integration, Training the Models. In *IEEE Big Data*.
[59] Anusha Kola, Harshal More, Sean Soderman, and Michael Gubanov. 2017. Generating Unified Famous Objects (UFOs) from the classified object tables. In *IEEE Big Data*.
[60] A. Laender, B. Ribeiro-Neto, A. Silva, and J. Teixeira. 2002. A Brief Survey of Web Data Extraction Tools. In *SIGMOD Record*. citeseer.ist.psu.edu/laender02brief.html
[61] Oliver Lehmberg, Dominique Ritze, Robert Meusel, and Christian Bizer. 2016. A Large Public Corpus of Web Tables containing Time and Context Metadata. In *WWW*, Jacqueline Bourdeau, Jim Hendler, Roger Nkambou, Ian Horrocks, and Ben Y. Zhao (Eds.).
[62] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and Searching Web Tables Using Entities, Types and Relationships. (2010).
[63] Hsuan-Tien Lin and Chih-Jen Lin. 2003. *A Study on Sigmoid Kernels for SVM and the Training of non-PSD Kernels by SMO-type Methods*. Technical Report. Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan.
[64] Reid McMurry, Patrick Lenehan, Samir Awasthi, and et al. 2021. Real-time analysis of a mass vaccination effort confirms the safety of FDA-authorized mRNA vaccines for COVID-19 from Moderna and Pfizer/BioNtech. *medRxiv*.
[65] Sutskever I. Chen K. Corrado G. S. Dean J Mikolov, T. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.
[66] Steven Ortiz, Caner Enbatan, Maksim Podkorytov, Dylan Soderman, and Michael Gubanov. 2017. Hybrid.JSON: High-velocity Parallel In-Memory Polystore JSON Ingest. In *IEEE Bigdata*.
[67] Sophie Pavia, Rituparna Khan, Anna Pyayt, and Michael Gubanov. 2022. Simplifying Access to Large-scale Structured Datasets by Meta-Profiling with Scalable Training Set Enrichment. In *SIGMOD*. ACM.
[68] Sophie Pavia, Nick Piraino, Kazi Islam, Anna Pyayt, and Michael Gubanov. 2022. Hybrid Metadata Classification in Large-scale Structured Datasets. *J. Data Intell.* 3, 4 (2022).
[69] Sophie Pavia, Montasir Shams, Rituparna Khan, Anna Pyayt, and Michael N. Gubanov. 2021. Learning Tabular Embeddings at Web Scale. In *Big Data*. IEEE.
[70] Thomas Pellissier Tanon, Gerhard Weikum, and Fabian Suchanek. 2020. YAGO 4: A Reason-able Knowledge Base. In *ESWC*, Andreas Harth, Sabrina Kirrane, Axel-Cyrille Ngonga Ngomo, Heiko Paulheim, Anisa Rula, Anna Lisa Gentile, Peter Haase, and Michael Cochez (Eds.).
[71] Maksim Podkorytov and Michael N. Gubanov. 2018. Hybrid.Poly: Performance Evaluation of Linear Algebra Analytical Extensions. In *IEEE Big Data*.
[72] Rajib Rana. 2016. Gated Recurrent Unit (GRU) for Emotion Classification from Noisy Speech.
[73] Abanoub Riad, Andrea Pokorná, and Sameh Attia et al. 2021. Prevalence of COVID-19 Vaccine Side Effects among Healthcare Workers in the Czech Republic, Vol. 10. JClMed.
[74] Montasir Shams, Sophie Pavia, Rituparna Khan, Anna Pyayt, and Michael N. Gubanov. 2021. Towards Unveiling Dark Web Structured Data. In *Big Data*. IEEE.
[75] Mark Simmons, Daniel Armstrong, Dylan Soderman, and Michael Gubanov. 2017. Hybrid.media: High Velocity Video Ingestion in an In-Memory Scalable Analytical Polystore. In *IEEE Bigdata*.
[76] Amit Singhal. 2012. Introducing the KG: Things, Not Strings. In *Google Blog*.
[77] Sean Soderman, Anusha Kola, Maksim Podkorytov, Michael Geyer, and Michael Gubanov. 2018. Hybrid.AI: A Learning Search Engine for Large-scale Structured Data. In *WWW*.
[78] Santiago Villasenor, Tom Nguyen, Anusha Kola, Sean Soderman, and Michael Gubanov. 2017. Scalable spam classifier for web tables. In *IEEE Big Data*.
[79] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex Wade, Kuansan Wang, Nancy Xin Ru Wang, Chris Wilhelm, Boya Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. CORD-19: The COVID-19 Open Research Dataset. In *arXiv, cs.DL 2004.10706*.
[80] Nasser Zalmout, Chenwei Zhang, Xian Li, Yan Liang, and Xin Luna Dong. 2021. All You Need to Know to Build a Product Knowledge Graph. In *SIGKDD*, Feida Zhu, Beng Chin Ooi, and Chunyan Miao (Eds.). ACM.