Virtual Generalist:

Modeling Co-morbidities in Synthea^{TM}

Organization: The Generalistas
Composed of data scientists and physicians from Microsoft Corp, Mercy Health and UC San Francisco.

Contact information:
Robert Horton, PhD, MS
Senior Data Scientist, Microsoft

John-Mark Agosta, PhD
Principal Data Scientist, Microsoft

Benjamin Dummitt, PhD
Lead Data Scientist, Mercy

Brandon DeShon, MS
Senior Data Scientist, Mercy

Katherine Gundling, MD FACP
Professor Emeritus, University of California, San Francisco

Jason Dausman, MD FACP FHM
Medical Director of Clinical Informatics, Mercy Hospital St. Louis

Challenge category: Enhancements to Synthea
Use Case: Complex Care
Video: https://youtu.be/HqB_thGSm1c
Github repository: https://github.com/rmhorton/virtual-generalist

Abstract

The Synthea^{TM} Patient Generator presents important new opportunities to create simulated data for health care and research. To achieve this purpose, however, Synthea must optimally reflect real medical encounters and disease states, which are often complex and nuanced. Most of the current Synthea modules function as “virtual specialists” that each focus on a relatively narrow scope of practice. To optimize Synthea’s functionality and application to the real world, a *Virtual Generalist* is required to generate a very wide range of complex combinations of conditions in statistically appropriate distributions. We developed hand-curated maps to compare observations made on real patients (coded in ICD10) to Synthea simulations (which use SNOMED concepts). Using these concept maps on a large clinical dataset, we were able to use machine learning (ML) on a scalable Spark platform to extract conditional probability tables (CPTs) that we incorporated into a Synthea simulation module. Our *Virtual Generalist CKD* module is notable in that it operates entirely by modifying attributes used by other modules, acting as a statistical supervisor to adjust population distributions. We validate the improved realism of the results using statistical measurements to compare distributions between simulated and real populations, and describe this validation approach in a detailed Appendix. Taking advantage of our current modeling of a comprehensive EMR dataset, we applied our existing methods to improvements in the Synthea simulation tools. Notebooks on our GitHub repository provide working examples other investigators can use to train and evaluate models on their own clinical data, to learn CPTs for other conditions. Importantly, we demonstrate our complex feature engineering and modeling approaches using Synthea data, which enables us to share working examples. Our tools, example module, and technical approaches help to democratize the use of ML models that can extract conditional probability tables, which are particularly suitable for use in Synthea's new `lookup_table_transition` state. Broad use of appropriate CPTs promises to add significant statistical sophistication to this simulation system. We provide a set of recommended design patterns for module authors that aim to increase Synthea modules’ amenability to the sort of statistical intervention that is possible with a Virtual Generalist. Together these tools and approaches empower developers to build simulation models with increased sophistication for complex disease conditions.

Introduction

Healthcare lags behind other industries in the level of sophistication of information technology, largely due to privacy restrictions on sharing data. Specifically, there is a paucity of shareable healthcare datasets suitable for machine learning (ML) applications.

Simulated data can only help fill this gap if it contains appropriate statistical relationships between attributes. Using ML to simulate data to train ML models seems oddly circular; if you understood the statistical relationships between attributes well enough to simulate data that could be used to train accurate ML models, you would need to have solved the ML problem already. However, simulated data does not need to capture the statistical relationships perfectly to be useful in practice. Even modestly realistic datasets can be quite useful for learning and experimenting with ML approaches, and to 'dry lab' various approaches that can subsequently be applied to real data.

Synthea^{TM} is an agent-based simulation platform supporting a growing collection of modules, many authored by contributing domain experts. These modules model incidence, progression, and treatment of clinical conditions. Most modules make use of publicly available health statistics, clinical guidelines, and patient care protocols in a human-understandable flowchart format, so the simulation logic is fully transparent. Each patient's lifetime is simulated to produce a population of patients. The resulting 'fully synthetic' medical records have proven useful for a variety of nonclinical settings, including education and many aspects of healthcare IT innovation [Walonoski_2020]. A particularly useful characteristic of the Synthea approach is that it produces *longitudinal* data, where trends may be observed for individual patients over time. This kind of data could potentially be used to develop and demonstrate time-dependent ML models, including time-series and probabilistic graphical models, if the simulated data were to incorporate appropriate statistical relationships among attributes.

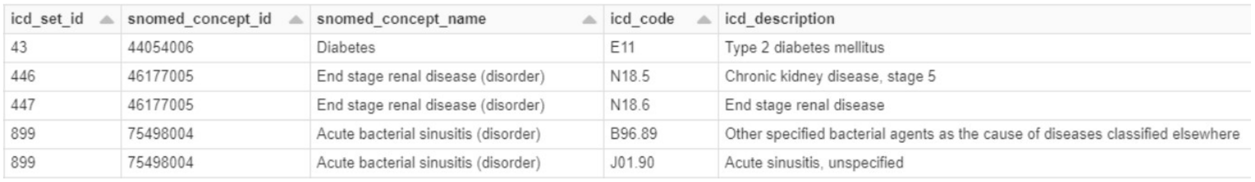
A great deal of data scientists' time is spent on various kinds of data manipulation, including (but not limited to) feature engineering. Feature engineering is the process of collecting information from various data sources to include in a dataset. For typical ML approaches, most datasets consist of single 'rectangular' tables containing one row per example, where one or more columns of the table contain *labels* (usually the outcome or category to be predicted), and other columns contain *features* which characterize the cases, and are used as inputs to predict the labels. The data scientist must decide how to represent these characteristics, and which characteristics to include; these decisions are implemented through feature engineering. The preferred way to share data manipulation approaches with other data scientists is with code accompanied by working examples. This requires sample data, and modestly realistic simulated data often fits the bill nicely.

The new 'lookup_table_transition' state in Synthea promises significantly improved power and simplicity for creating Synthea modules. These tables make it possible to replace complicated networks of dependencies between states in Synthea modules with simple, clean table lookups. Moving the complexity from the state diagrams to tables will greatly facilitate the development of much more sophisticated conditional logic in Synthea modules. This is a new feature, and the support is not complete; for example, the JSON code must be modified slightly for display in the Module Builder, compared to the format required to run in Synthea.

But how do you decide which conditions to include in a conditional probability table, and where do you get the probabilities? Here we demonstrate a data-driven approach using machine learning of Bayesian belief networks [Pearl 1988] to model conditional probabilities and generate the kinds of tables we need, then deploy these tables in our Virtual Generalist module in Synthea. This module is unique in that its only effects are to alter the attributes used by another module. This results in a more realistic distribution of conditions in the simulated population without modifying the code of the other module.

Methods

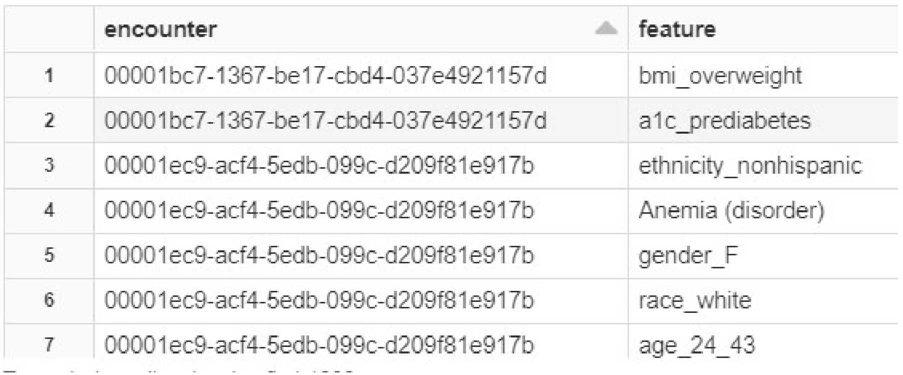
Details of feature engineering and modeling approaches can be found in the notebooks on our github repository at https://github.com/rmhorton/virtual-generalist.



icd_set_id	snomed_concept_id	snomed_concept_name	icd_code	icd_description
43	44054006	Diabetes	E11	Type 2 diabetes mellitus
446	46177005	End stage renal disease (disorder)	N18.5	Chronic kidney disease, stage 5
447	46177005	End stage renal disease (disorder)	N18.6	End stage renal disease
899	75498004	Acute bacterial sinusitis (disorder)	B96.89	Other specified bacterial agents as the cause of diseases classified elsewhere
899	75498004	Acute bacterial sinusitis (disorder)	J01.90	Acute sinusitis, unspecified

Figure 1: Mapping ICD10 codes to SNOMED concepts.

Concept mapping: Figure 1 shows examples of our hand-curated mapping rules. The first row (set_id 43) shows a truncated ICD10 pattern; any code starting with 'E11', including all of the more detailed codes with additional digits, will map to the 'Diabetes' concept. We chose not to create a separate concept for Type 2 Diabetes because 'Diabetes' is the concept currently used in Synthea, and it is apparently just Type 2. The next two lines show that ICD10 codes N18.5 and N18.6 both map to 'End stage renal disease'; the Synthea Metabolic Syndrome disease module treats stage 5 as end stage, and we do the same. The last two rows show a two-part rule; a patient must have both acute sinusitis and bacterial agents before we decide they have acute bacterial sinusitis.



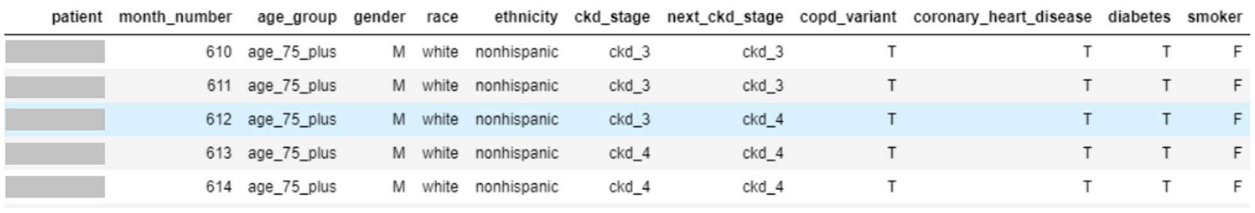
	encounter	feature
1	00001bc7-1367-be17-cbd4-037e4921157d	bmi_overweight
2	00001bc7-1367-be17-cbd4-037e4921157d	a1c_prediabetes
3	00001ec9-acf4-5edb-099c-d209f81e917b	ethnicity_nonhispanic
4	00001ec9-acf4-5edb-099c-d209f81e917b	Anemia (disorder)
5	00001ec9-acf4-5edb-099c-d209f81e917b	gender_F
6	00001ec9-acf4-5edb-099c-d209f81e917b	race_white
7	00001ec9-acf4-5edb-099c-d209f81e917b	age_24_43

Figure 2:Encounter features in long format.

Encounter-level data: A panel of categorical features (is computed for each patient encounter (BMI category, age group, Hemoglobin A1c range, patient age at the time of the encounter, etc) and the SNOMED concepts associated with that encounter are used as features as well. This figure shows features engineered from Synthea data (see the notebook "Synthea_population_statistics" on our github repository for details), so we are free to show you the encounter IDs. Of course we use this same approach on real data to learn realistic conditional probabilities.

Co-occurrence graphs: For every pair of features in the table shown in Figure 2 (age group, SNOMED concept, etc.) we compute a variety of statistics quantifying how often those two features appear in the same encounter. For example, the metric 'confidence' is the fraction of encounters having the first feature that also have the second feature, and the metric 'lift' is the confidence normalized by overall prevalence of the second feature. These pairwise relationship metrics become edge weights in a graph where the features are nodes. These graphs are visualized using the 'vis.js' javascript library.

Encounter-level Bayes Network: The encounter feature table is pivoted to a wide format, so that there is one row per encounter and one column per feature, and used as input to the `bnlearn` package in R [Nagarajan 2013] to learn network structure, then compute conditional probabilities.



patient	month_number	age_group	gender	race	ethnicity	ckd_stage	next_ckd_stage	copd_variant	coronary_heart_disease	diabetes	smoker
[redacted]	610	age_75_plus	M	white	nonhispanic	ckd_3	ckd_3	T	T	T	F
[redacted]	611	age_75_plus	M	white	nonhispanic	ckd_3	ckd_3	T	T	T	F
[redacted]	612	age_75_plus	M	white	nonhispanic	ckd_3	ckd_4	T	T	T	F
[redacted]	613	age_75_plus	M	white	nonhispanic	ckd_4	ckd_4	T	T	T	F
[redacted]	614	age_75_plus	M	white	nonhispanic	ckd_4	ckd_4	T	T	T	F

Figure 3: Monthly patient data in wide format, for training dynamic Bayes net on CKD stages. Patient IDs are redacted, but all rows shown here apply to the same patient. Note the transition from stage 'ckd_3' to 'ckd_4' on the third row.

Monthly timeseries data for CKD stages: Each row in this dataset shows the features for one patient for one month, and each month of the patient's history gets a row. This dataset uses a much smaller set of features than then encounter level data, and includes a column for 'ckd_stage', as well as a column showing what stage that patient will be in the next month ('next_ckd_stage' is just 'ckd_stage' slid forward one row.) Missing values are filled forward from the last known value; if there is no known value for ckd stage it is assumed to be zero.

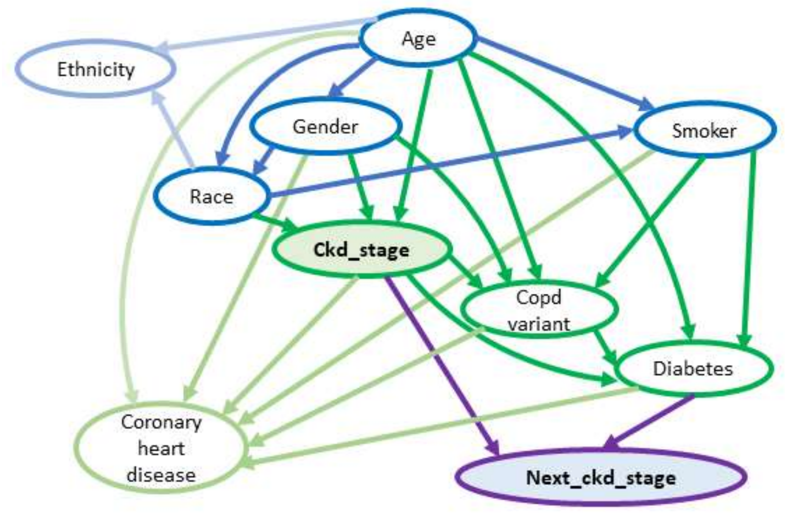


Figure 4: Structure of the dynamic Bayes net model of stage transitions in CKD. Each node in this graph has an accompanying conditional probability table (CPT). The CPT from the 'next_ckd_stage' node was incorporated into our Synthea module.

Dynamic Bayes network for CKD stage progression: When the monthly timeseries data is used to build a Bayes network, it learns the dependencies between 'next_ckd_stage' and the other features. Building the Bayes network determines feature importance for the transition model. Here the effects of all variables on *CKD stage transition* are mediated solely through *Diabetes*, which itself is influenced by *Age*, *Smoker*, *Copd Variant*, and *CKD stage*. Both *Ethnicity* and *Coronary heart disease* are irrelevant to *Next_ckd_stage*, given the rest. The features in this dataset are all closely tied to Synthea attributes, which enables the lookup_table_transition module to use attribute values as inputs when the simulation runs.

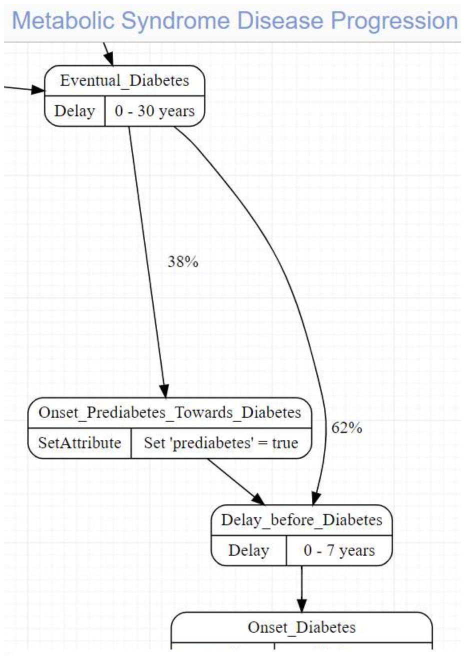


Figure 5: Diabetes and CKD are predestined after patient reaches the age of 18.

Results

We have developed a process for learning conditional probability lookup tables from data at scale. We demonstrate the use of these learned tables in `lookup_table_transition` states for chronic kidney disease (CKD) in Synthea, and show that this leads to more realistic statistical distribution of disease stages in the resulting simulated population.

Figure 5 shows the section of the Metabolic Syndrome disease progression module that determines whether a simulated patient will be given diabetes. This decision is made just after the patient turns 18, and it leads to a delay of up to 37 years before the disease manifests. This means patients are destined to get diabetes that leads to CKD, and nothing that happens in the simulation after this point can alter that decision.

However, once a patient enters the disease progression loop, the stage of their chronic kidney disease is maintained in the `ckd` attribute, which is a number between 0 (no disease) and 5 (end stage disease). The loop runs once per month, and in every cycle there are fixed probabilities for progressing from the given stage to the next.

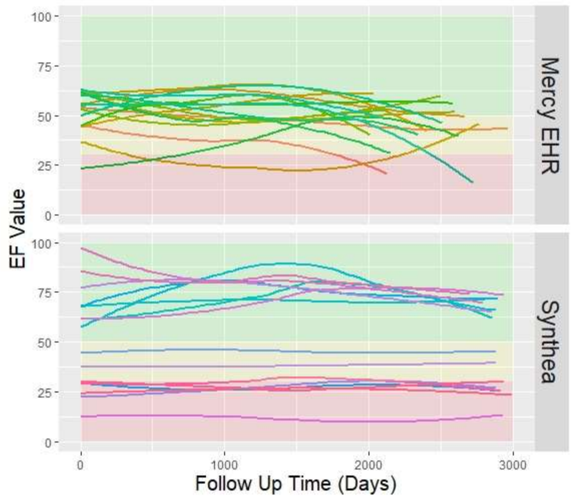
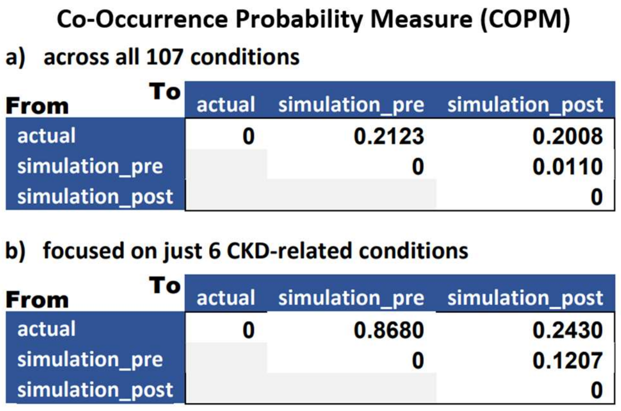


Figure 6: Time distribution of Left Ventricular Ejection Fraction in real vs. simulated patients.

This disease progression loop has sections for other components of metabolic syndrome, including peripheral neuropathy and retinopathy; we chose to prioritize CKD because we can map the disease states to ICD10 codes, which let us model disease progression on real data.

Other modules do not permit intervention of this sort; we examine the Congestive Heart Failure module as a prime example. Left Ventricular Ejection Fraction is a marker of disease progression in this condition, but it is not used that way in Synthea. Figure 5 compares measurements over time for several individual patients, with real patients from the Mercy database on the top panel, and simulated patients from Synthea on the bottom.



Co-Occurrence Probability Measure (COPM)

a) across all 107 conditions

From \ To	actual	simulation_pre	simulation_post
actual	0	0.2123	0.2008
simulation_pre		0	0.0110
simulation_post			0

b) focused on just 6 CKD-related conditions

From \ To	actual	simulation_pre	simulation_post
actual	0	0.8680	0.2430
simulation_pre		0	0.1207
simulation_post			0

Figure 7: Model co-occurrence probability measure mean values between simulated and real patient populations, before and after incorporating the Virtual Generalist CKD module.

As shown In Figure 7, COPM values computed on simulated populations (100 thousand patients each) before and after incorporating the Virtual Generalist CKD module. The upper table considers a broad panel of 107 variables, while the lower table considers just a subset of 6 of those variables affected by the improvement. In both cases COPM decreases after our intervention, indicating the simulation more closely approximates the clinical EMR data from which the module improvements are derived. As expected, the measured improvement is more pronounced when considering just the variables most directly affected by the module, but the improvement is noticeable even in a very general evaluation.

Our code runs on the Azure Databricks Spark platform, and is scalable to extremely large datasets. It is provided in the form of Databricks notebooks that use SQL, Python, and R code cells to perform the required data manipulation, train the Bayes network models, and extract the desired lookup tables. We also generate interactive graph visualizations illustrating the co-occurrence relationships between features in the data.

Discussion

Recent progress in machine learning is supported to a large degree by freely shareable datasets, which are used to demonstrate, communicate, and evaluate new ML approaches. There is a paucity of such material for medical applications, largely due to privacy and confidentiality concerns, and simulation promises to help fill this gap in useful ways.

It may seem circular to try to learn lookup tables to use in a simulation from data that was generated by a simulation. The point is to make it easier to show others how to learn these lookup tables, so that they can go on to adapt the process to their own real data, and learn new lookup tables. The process of manipulating and formatting the data to prepare it for machine learning is nontrivial, and it is difficult to overstate the practical importance of being able to share working examples of code together with the data it operates on. Providing shareable data is one of the most important uses of health data simulation as far as software development is concerned, and this includes development of AI applications.

We show the power of our approach by developing a Synthea module using `lookup table transition` states. The 'virtual_generalist_ckd' module modifies the transition probabilities of going from one stage of chronic kidney disease (CKD) to another. The state transitions were learned using a dynamic Bayes net trained on a dataset where each row is a time slice (one month in this case, to match the granularity used in the original disease progression module). One feature of these time slices is the current CKD stage, and another is the stage of the patient's disease in the subsequent month. Additional features include age, gender, and co-morbidities (hypertension, etc.).

We also use these conditional probabilities to provide a statistical overview of the relationships between observable conditions. We can use this perspective to make population-level comparisons, both between real and simulated data and between simulated datasets before and after our module is included, as described in detail in Appendix A.

Our code, together with working examples of the complex, scalable feature engineering and modeling process that we used to develop these models and extract the conditional probability tables, are available from our GitHub repository. Working examples require sample data, which we conveniently obtained from Synthea. We regard this application of synthetic data, which enables shareable working examples of data science approaches, as an extremely important use case for Synthea.

Recommended design patterns for module authors

No Delay: try to avoid using long delay transitions after a decision is made to give a patient a disease.
Mechanism of Progress: Employ attributes that serve as mechanistic determinants of progression within a continuous loop. This makes it possible for other modules to affect disease progression my altering the values of these attributes.
Separate treatment**:** keep the modeling of disease incidence and progression separate from patient care, in a different module if possible (see the Metabolic Syndrome disease progression and standards of care modules for an example).

Donation of Prize Money

To avoid any potential conflicts of interest with our day jobs, we request that in the event this team wins any prize money in this contest it be donated to:

Sustainable Harvest International
177 Huntington Ave Ste 1703 #23701
Boston, MA 02115 USA

References

Brady E, Nielsen MW, Andersen JP, Oertelt-Prigione S. Lack of consideration of sex and gender in COVID-19 clinical studies. Nat Commun. 2021 Jul 6;12(1):4015. doi: 10.1038/s41467-021-24265-8. PMID: 34230477.

Choi E, Biswal S, Malin B, Duke J, Stewart WF, and Sun J. 2017. Generating multi-label discrete patient records using generative adversarial networks. In Machine learning for healthcare conference. PMLR, 286–305.

Lu PH, Wang PC, and Yu CM. 2019. Empirical evaluation on synthetic data generation with generative adversarial network. In Proceedings of the 9th International Conference on Web Intelligence, Mining and Semantics. 1–6

Marini, S, Trifoglio E, Barbarini N, Sambo F, Di Camillo B, Malovini A, Manfrini M, Cobelli C, Bellazzi R., "A Dynamic Bayesian Network model for long-term simulation of clinical complications in type 1 diabetes," *Journal of Biomedical Informatics,* Volume 57, (2015), pp 369-376.

Nagarajan R, Scutari M and Lèbre S. Bayesian Networks in R with Applications in Systems Biology. Use R!, Vol. 48, Springer (US), 2013.

Pearl J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann, 1988.

Pearl J and MacKenzie D. The Book of Why - the New Science of Cause and Effect. Basic Books, New Yory NY, 2018, p 197.

Templ M, Meindl B, Kowarik A, and Dupriez O. 2017. Simulation of synthetic complex data: The R package simPop. Journal of Statistical Software 79, 10 (2017), 1–38

Walonoski J , Klaus S, Granger E, Hall D, Gregorowicz A, Neyarapally G, Watson A, and Eastman J. 2020. Synthea™ Novel coronavirus (COVID-19) model and synthetic data set. Intelligence-based medicine 1 (2020), 100007.

Walonoski J, Kramer M, Nichols J, Quina A, Moesel C, Hall D, Duffett C, Dube K, Gallagher T, and McLachlan S. 2018. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record.