UNIVERSITY OF COPENHAGEN

Københavns Universitet

Using sequences of life-events to predict human lives

Savcisens, Germans; Eliassi-Rad, Tina; Hansen, Lars Kai; Mortensen, Laust Hvas; Lilleholt, Lau; Rogers, Anna; Zettler, Ingo; Lehmann, Sune

Published in:
Nature Computational Science

DOI:
10.1038/s43588-023-00573-5

Publication date:
2024

Document version
Early version, also known as pre-print

Citation for published version (APA):
Savcisens, G., Eliassi-Rad, T., Hansen, L. K., Mortensen, L. H., Lilleholt, L., Rogers, A., Zettler, I., & Lehmann, S. (2024). Using sequences of life-events to predict human lives. *Nature Computational Science*, *4*, 43–56. https://doi.org/10.1038/s43588-023-00573-5

Download date: 12. Mar. 2026

arXiv:2306.03009v1 [stat.ML] 5 Jun 2023

Using Sequences of Life-events to Predict Human Lives

Germans Savcisens, Tina Eliassi-Rad, Lars Kai Hansen, Laust Hvas Mortensen, Lau Lilleholt, Anna Rogers, Ingo Zettler, and Sune Lehmann

June 6, 2023

Abstract

Over the past decade, machine learning has revolutionized computers' ability to analyze text through flexible computational models [1]. Due to their structural similarity to written language, transformer-based architectures [2] have also shown promise as tools to make sense of a range of multi-variate sequences from protein-structures [3, 4], music [5, 6], electronic health records [7] to weather-forecasts [8, 9]. We can also represent human lives in a way that shares this structural similarity to language [10]. From one perspective, lives are simply sequences of events: People are born, visit the pediatrician, start school, move to a new location, get married, and so on. Here, we exploit this similarity to adapt innovations from natural language processing to examine the evolution and predictability of human lives based on detailed event sequences. We do this by drawing on arguably the most comprehensive registry data in existence, available for an entire nation of more than six million individuals across decades [11, 12, 13, 14]. Our data include information about life-events related to health, education, occupation, income, address, and working hours, recorded with day-to-day resolution. We create embeddings of life-events in a single vector space showing that this embedding space is robust and highly structured. Our models allow us to predict diverse outcomes ranging from early mortality to personality nuances, outperforming state-of-the-art models by a wide margin. Using methods for interpreting deep learning models, we probe the algorithm to understand the factors that enable our predictions. Our framework allows researchers to identify new potential mechanisms that impact life outcomes and associated possibilities for personalized interventions.

1 Introduction

We live in the age of algorithm-driven prediction of human behavior. The predictions range from the global and population level, where societies allocate vast resources to predicting phenomena such as global warming [15] or the spread of infectious diseases [16], all the way to the constant flow of individual micro-predictions that shape our reality and behavior as

we use social media [17]. When it comes to individual life outcomes, however, the picture is more complex: While it is known that socio-demographic factors play an important role in human lives [18], a collaboration of 160 teams independently analyzing in small groups a comprehensive birth cohort dataset collected over more than 15 years has recently argued that the predictions are typically not accurate, suggesting practical upper limits for predictions of life outcomes [19].

Here, we find that with highly detailed data, a different picture of individual-level predictability emerges. Drawing on a unique dataset consisting of detailed individual-level day-by-day records [13, 14], describing the 6 million inhabitants of Denmark, spanning a 10-year interval, we show that accurate individual predictions are indeed possible. Our dataset includes a host of indicators, such as health, professional occupation and affiliation, income level, residency, working hours, and education (Methods, Sec. 4.2).

The central reason we are currently experiencing this age of human prediction is the advent of massive datasets and powerful machine learning algorithms [20, 21, 22]. Over the past decade, machine learning has revolutionized image and text processing fields by accessing ever larger datasets that have enabled increasingly complex models [23, 24, 25]. Language processing has evolved particularly rapidly, and transformer architectures have proven successful at capturing complex patterns in massive and unstructured sequences of words [26, 27, 28]. While these models originated in natural language processing, their ability to capture structure in human language generalizes to other sequences [3, 4, 5, 6, 7, 8, 9, 29, 30], which share properties with language, e.g., that sequence ordering is essential, and elements in the sequence can have meaning on many different levels. Importantly, due to the absence of large-scale data, transformer models have not been applied to multi-modal socio-economic data outside the industry.

Our dataset changes this. The sheer scale of our dataset allows us to construct sequence-level representations of individual human life-trajectories, which detail how each person moves through time. We can observe how individual lives evolve in the space of diverse types of events (information about a heart attack is mixed with salary increases or information about moving from an urban to a rural area). The time resolution within each sequence and the total number of sequences are large enough that we can meaningfully apply transformer-based models to make predictions of life outcomes. This means that representation learning can be applied to an entirely new domain to develop a new understanding of the evolution and predictability of human lives. Specifically, we adopt a BERT-like architecture [31] to predict two very different aspects of human lives: time of death and personality nuances (additional predictions in SI: Emigration Tasks). We find that our model can accurately predict these outcomes, in the case of early mortality, outperforming current state-of-the-art methods by $\sim 11\%$, see *Results*.

To make these accurate predictions, our model relies on a single common embedding space for all events in the life-trajectories. Just as embedding spaces in language models can be studied to provide a novel understanding of human languages [32, 33], we can study the concept embedding space to reveal non-trivial interactions between life-events. Below, we provide insight into the resulting *concept-space* of life-events and demonstrate the robustness and interpretability of this space and the model itself. Transformer-based models also produce an embedding of individuals (the analogy in a language representation is a vector summarizing an entire text). Using explainability tools such as saliency maps [34, 35] and concept activation vectors (TCAV) [36], we show that the person-summaries are also meaningful and hold the potential to serve as a *behavioural phenotype* which can improve other individual-level prediction tasks, for example, to augment analyses of medical images [37]. Our work has important societal and ethical implications, which we outline in the Discussion as well as in Methods, Sec. 4.1, and SI: Model Card.

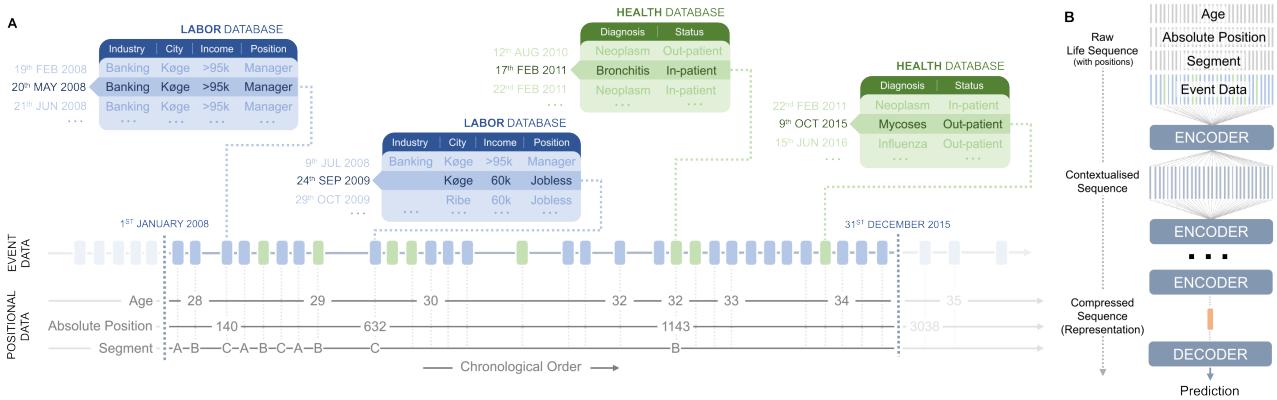

Figure 1: A schematic individual-level data representation for the `life2vec` model. (A) We organize socio-economic and health data from the Danish national registers from 1st January 2008 until 31st December 2015 into a single chronologically ordered *life-sequence*. Each database entry becomes an event in the sequence, where an event has associated positional and contextual data. The contextual data include variables associated with the entry (e.g., industry, city, income, job type). The positional data includes the person's age (expressed in full years), absolute position (number of days since 1st January 2008), and segment (alternating sequence of three elements). The raw life-sequence is then passed to the model described in panel (B). The model consists of multiple stacked encoders. The first encoder combines contextual and positional information to produce a contextual representation of each life event. The following encoders output deep contextual representations of each life event (considering the overall content of the life-sequence). The final encoder layer fuses the representations of life-events to produce the representation of a life-sequence. The decoder uses the latter to make predictions.

2 Results

2.1 Life-events and Life-sequences

Life-sequences for millions of individuals based on rich data. In the following, we represent the progression of individual lives as *life-sequences* (see Fig. 1). The life-sequences are constructed based on labor and health records from Danish national registers [13, 14], which contain highly detailed data on work, residence, health, and education for all ~ 6 million Danish citizens. Our *labor* dataset [11] includes records about income, such as salary, scholarship, job-type [38], industry [39], social benefits, etc. The *health* dataset [12, 13] includes records about initial visits to healthcare professionals or hospitals, accompanied by the diagnosis, patient type, and urgency (encoded according to the ICD-10 system [40], SI: Specification of features and their sources). Life-sequence evolve over time and provide rich information about life-events with high temporal resolution.

We use a simple symbolic language to encode the rich data. The raw stream of complex multi-source temporal data poses significant methodological challenges, such as irregular sampling rates, sparsity of data, complex interactions between features, and a high number of dimensions [41]. Classical methods for time series analysis (e.g., support vector machines, ARIMA) [42, 43] become cumbersome because they are challenging to scale, inflexible, and require a considerable amount of data preprocessing to extract useful features. Using transformer methods allows us to avoid hand-crafted features and instead encode the data in a way that exploits the similarity to language [43]. Specifically, in our case, each category of discrete features and discretized continuous features form a *vocabulary*. This vocabulary – along with an encoding of time – allows us to represent each life-event (including its detailed qualifying information) as a *sentence* comprised of synthetic words, or *concept tokens*. We attach two temporal indicators to every event. One that specifies the individual's age at the time of the event and one that captures absolute time, see Fig. 1.

Thus, our synthetic language can capture information along the lines of "In September 2020, Francisco received twenty thousand Danish kroner as a guard at a castle in Elsinore" or "During her third year at secondary boarding school, Hermione followed five elective classes". In this sense, the progression of a person's life is represented as a string of such sentences that together form individual *life-sequences*. Our approach allows us to encode a wide range of detailed information about events in individual lives without sacrificing the content and structure of the raw data.

2.2 The `life2vec` model

We use transformer models to form compact representations of individual lives. We call our deep learning model `life2vec`. The `life2vec` model is based on a transformer-architecture [31, 30, 44, 45, 46, 47, 48, 49, 50, 51]. Transformers are well suited for representing life-sequences due to their ability to compress contextual information [52, 53] and take into account temporal and positional information [5, 54].

The training of the `life2vec` consists of two stages. We first train the model by simultaneously using (1) a Masked Language Modeling (MLM) task that forces the model to use token representations and contextual information [31] and (2) a Sequence Ordering Prediction (SOP) task that focuses on the temporal coherence of the sequence [55] (Methods, Sec.: 4.4). The pre-training creates a concept space and teaches the model patterns in the structure of life-sequences, which we discuss below.

Next, to create compact representations of individual life-sequences, the model performs a classification task (Methods, Sec.: 4.4). The person-summaries the model learns in this last step is *conditional* on the classification task; it identifies and compresses patterns that maximize the certainty around a given downstream task [56]. For example, when we ask the model to predict a person's personality nuances, the person embedding space will be structured around key dimensions that contribute to personality.

2.3 Accurate predictions across diverse domains

The first critical test of any model is predictive performance. Here, `life2vec` outperforms the state-of-the-art while simultaneously being able to perform classification in very different domains. We test our framework on two distinct tasks.

Predicting early mortality. We estimate the likelihood of a person surviving the following four years after 1st January 2016. This is an oft-used task within statistical modeling [57]. Further, mortality prediction is closely related to other health-prediction tasks and therefore requires `life2vec` to model the progression of individual health-sequences as well as labor history to predict the right outcome successfully. Specifically, given a sequence representation, `life2vec` infers the likelihood of a person surviving the four years following the end of our sequences (1st January 2016). We focus on making predictions for a young cohort of people consisting of individuals who are 30-55 years old, where mortality is challenging to predict.

This prediction task has an additional level of complexity as data contains people with unknown outcomes (i.e., emigrants and missing individuals). We account for this issue by applying positive-unlabeled learning [58, 59], which gives us a robust loss function for training,

as well as a corrected performance metric for the model evaluation.

The performance of `life2vec` in relation to a range of baseline models [60]—actuarial life tables, logistic regression, feed-forward neural networks, and recurrent neural networks, is shown in Fig. 2 and summarized in Tab. A8.

We illustrate the performance of models using the Corrected Matthews correlation coefficient, C-MCC [61, 62] (Methods, Sec.: 4.6.1) that adjusts the MCC value due to the presence of unlabeled samples. With the median C-MCC Score of 0.41 (95% CI [0.40, 0.42]), `life2vec` outperforms the baselines by 11% (see Fig. 2); note that increasing the size of RNN models does not improve their performance. Fig. 2.D also breaks down performance for various sub-groups: intersectional groups based on age and sex, as well as groups based on the sequence length (SI: Model Card).

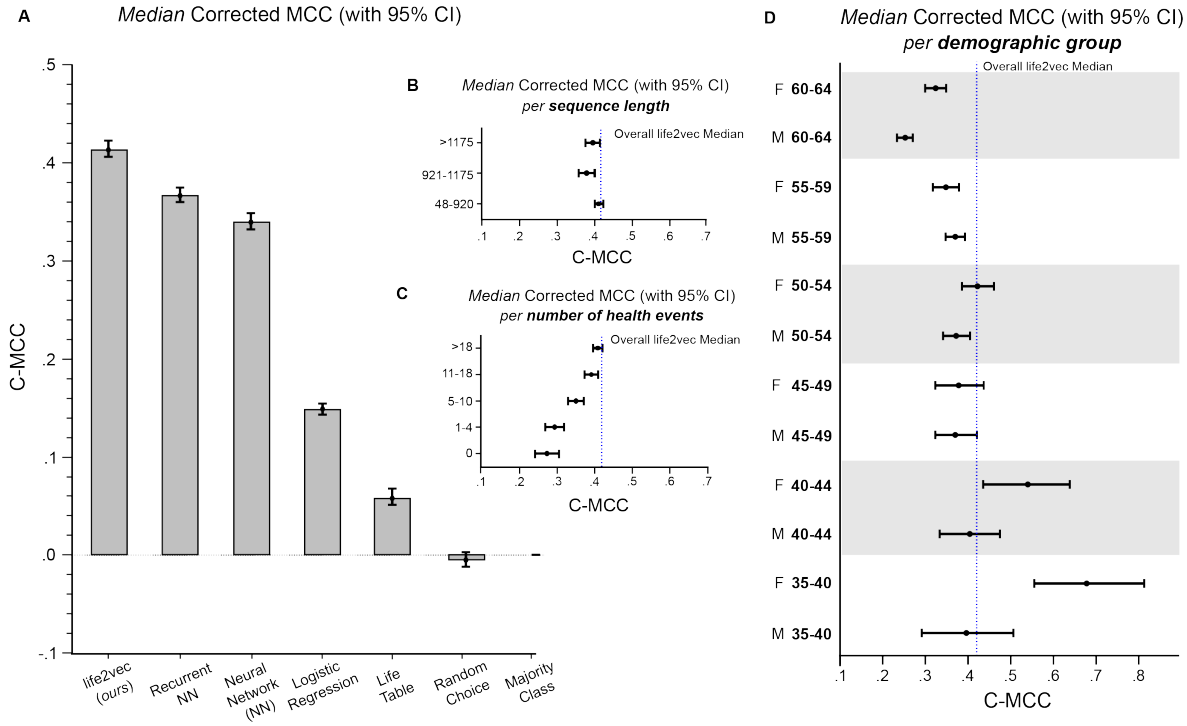

Figure 2: Performance of models on the Mortality Prediction Task quantified with the Median Corrected Matthews correlation coefficient (C-MCC) [62] with 95% CI. (**A**) Comparison of `life2vec` performance to baselines (**B-D**) Performance of `life2vec` model on different cohorts of the population. (**B**) Performance of `life2vec` per sequence length. We can see that sequence length does not affect the performance. (**C**) Performance of `life2vec` based on the number of health events in a sequence. The model performs better on cohorts with a higher number of health events. (**D**) Performance of `life2vec` per inter-sectional groups (based on age group and sex).

In terms of age and gender, the model performs better on a younger cohort of the population and on a cohort of females. Further, sequence length (i.e., a proxy for a number of life-events in a sequence) does not have a significant impact on the performance of a model (Fig. 2B).

Predicting personality nuances. Death as a prediction target is well-defined and eminently measurable. To test the versatility of `life2vec`, we now predict *personality nuances*, an outcome at the other end of the measurement spectrum, something which is internal to an individual and typically measurable through questionnaires. In spite of the difficulty in measurement, personality is an important feature that shapes people's thoughts, feelings, and behavior and predicts life outcomes [63]. Specifically, we focus on personality nuances in the domain of the Introversion-Extraversion dimension (for simplicity, Extraversion in what follows) because the corresponding personality nuances are part of virtually all comprehensive models of the basic personality structure that have emerged (in the Western world) over the last century, including the Big Five [64] and HEXACO [65] frameworks, but also Eysenck's [66] and Jung's [67] personality models. We align the prediction of personality nuances by `life2vec` with recent research that highlights the advantages of personality nuances (i.e., responses to specific personality questionnaire items) over broader summarizing (i.e., responses across items) personality 'facets' (e.g., Extraversion-Social Self-esteem) and 'domains' (e.g., Extraversion) in terms of associations with life outcomes [68, 69, 70]. As our dataset, we draw on data collected for a large and largely representative group of individuals in 'The Danish Personality and Social Behavior Panel' (POSAP) study [71] (see Methods Sec. 4.2). We randomly pick one item (personality nuance) per Extraversion facet and predict individual-level answers.

Fig. 3 shows that applying `life2vec` to life-sequences not only allows us to predict early mortality but is versatile enough also to capture personality nuances (see Methods Sec. 4.4.2). `life2vec` has better scores than RNN on all items, but the difference is only statistically significant on Items 2 and 3 (see Fig. 3 for item wording). The fact that an RNN trained for this specific task is also able to extract a signal around personality underscores that – while transformer models are powerful – a large part of what makes `life2vec` so versatile is the dataset itself.

We have illustrated `life2vec`'s versatility with further prediction tasks (SI: Emigration Task).

2.4 Concept Space: Understanding relations between concepts

The building blocks of `life2vec` are the concept tokens of our synthetic language. A key novelty of our approach is that the algorithm learns a single joint multidimensional space that contains all events that can occur in human life. We start our exploration of this space with a visualization.

The global view. In Fig. 4, the original 280-dimensional concepts are projected onto a two-dimensional manifold with the use of PaCMAP [72], that preserves the local and global structures of the high-dimension space. PaCMAP constructs the graph consisting of three types of edges – that connect neighbors, mid-near pairs, and further pairs. These edges

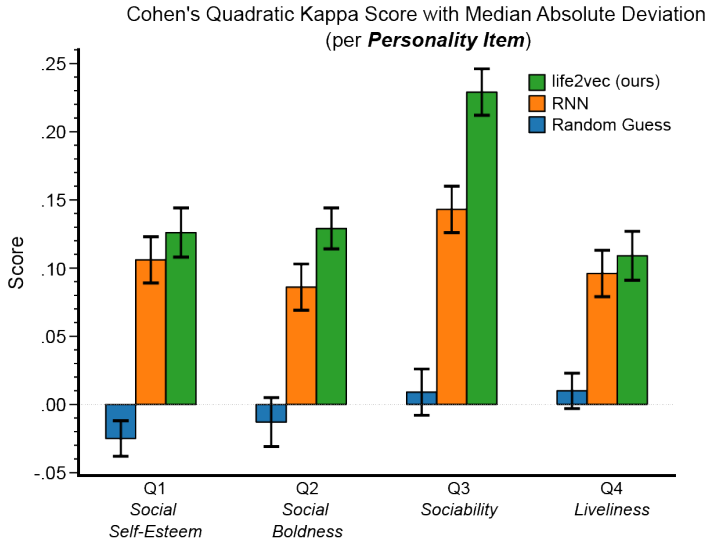

Figure 3: Performance Evaluation for the Personality Nuances Task. We display Cohen's Quadratic Kappa score for each item separately for Random Guess, RNN, and `life2vec` model. The error bars indicate the Median Absolute Deviation. The question wordings are as follows. Q1 (Social Self-esteem): "I feel reasonably satisfied with myself overall". Q2 (Social Boldness): "When I'm in a group of people, I'm often the one who speaks on behalf of the group". Q3 (Sociability): "I prefer jobs that involve active social interaction to those that involve working alone" Q4 (Liveliness): "On most days, I feel cheerful and optimistic".

define how forces of attraction and repulsion should move points along the two-dimensional manifold [72].

Here, each concept is colored according to its type. This coloring makes it clear that the overall structure is organized according to the key concepts of the synthetic language: health, job type, municipality, etc., but with interesting sub-divisions, separating a birth year, income, social status, and other key demographic pieces of information. The structure of this space is highly robust and emerges reliably under a range of conditions (see Methods Sec. 4.6).

The fine structure of concept space is meaningful. Digging deeper than the global layout, we find that the model has learned intricate associations between nearby concepts. We investigate these local structures via neighbor analysis, which draws on the cosine distance between concepts in the original high-dimensional representations as a similarity measure. A key place to consider is the cluster formed by income (dark blue points in Fig. 4). What the model sees is 100 concept tokens, each describing a level of income – but before training, it has no *a priori* idea of what each one means. It is simply an arbitrary string of text among other strings, but from training on the life-sequences, the model not only learns that income is different from other concepts (the dark blue points are isolated), but it also perfectly sorts the 100 levels. The blue curve starts with the token corresponding to the first percentile salaries and organizes them up to the 100th. Thus, the concepts most similar to the 59th percentile of income are the 58th and the 60th. Similarly, for birth years (light blue in Fig. 4):

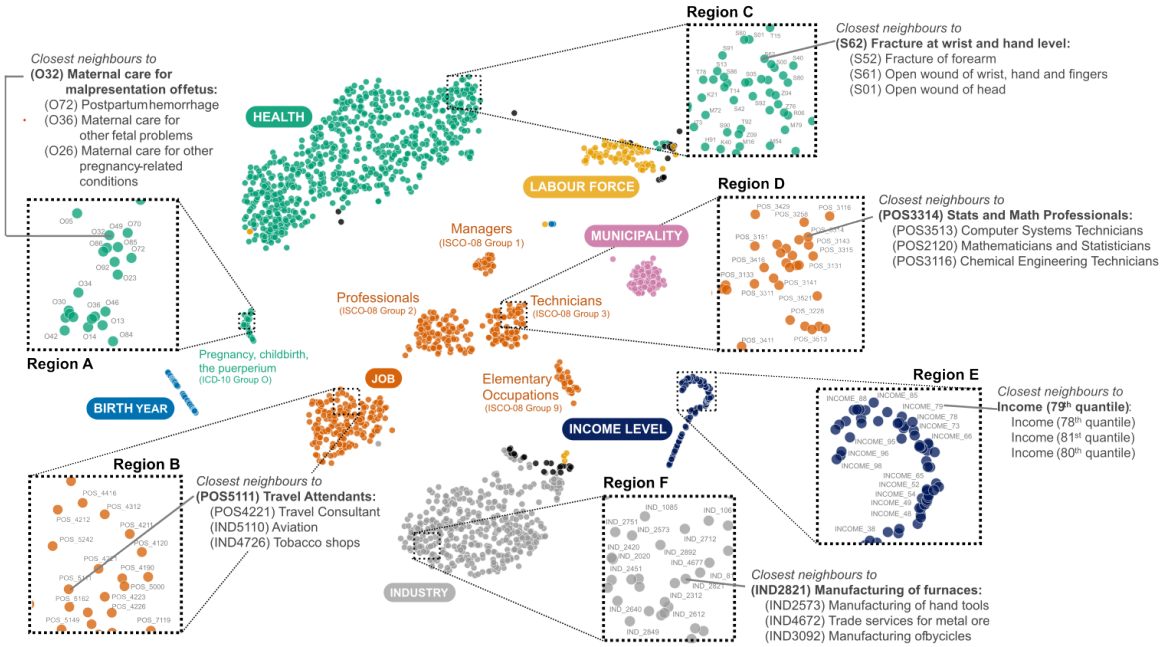

Figure 4: Two-dimensional projection of the concept space (using the PaCMAP [72]). Each point corresponds to a concept token in the vocabulary. Points are colored based on the concept types (several types are omitted - black points). Each region provides a closer look at several parts of the concept space. You can also see the top three closest neighbors for selected tokens (based on the cosine distance). (**A**) Diagnoses related to Pregnancy, childbirth, and the puerperium in ICD-10 [40]. (**B**) Job concepts related to Service and Sales Workers (corresponds to Job Category 5 of ISCO-08 [38]). (**C**) Injury-related diagnoses in ICD-10 [40]. (**D**) Job concepts related to Technicians and Associate Professionals (corresponds to Job Category 3 of ISCO-08 [38]). (**E**) Income-related concepts. `life2vec` arranges these concepts in increasing ordinal order. (**F**) Concepts related to the manufacturing industry in DB07 [39].

the closest concepts to the birth year 1963 are 1962 and 1964, and so on.

The health-type cluster (green points in Fig. 4) has a solid local structure. Diagnoses belonging to the same ICD-10 [40] chapters cluster according to their chapter. For example, the concept 'malignant neoplasm of stomach' (C16 in ICD-10) is surrounded by other C-Chapter concepts, such as 'malignant neoplasm of lungs' (C34) and 'malignant neoplasm of colon' (C18). As shown in Fig. 4A, one of the clearly separated health-clusters relates to pregnancies and childbirth diagnoses (i.e., O-Chapter concepts).

The concepts of professional occupation also cluster into smaller groups. These groups roughly correspond to the Major Groups of the International Standard Classification of Occupations (ISCO-08) [38]. Clearly defined clusters exist for 1st (Managerial and Executive

Positions), 2nd (Professionals), 3rd (Technicians and Associate Professionals), and 9th (Elementary Occupations) groups.

Not all concept tokens are surrounded by tokens of the same category, but even in these cases, the neighborhoods are meaningful. In Fig. 4B job-concept of a 'travel agent' is surrounded by the job-concept of a 'travel consultant' and an industry-concept of Aviation. When the model does mix up ICD-10 codes, the 'mistakes' are meaningful. For example, the concept of Z95 (Presence of cardiac and vascular implants and grafts) is surrounded by concepts corresponding to ICD-10 Chapter I [40], for example, I42 (Cardiomyopathy), I50 (Heart failure), and I25 (Chronic ischemic heart disease). The model's ability to group similar concepts that are not necessarily close in the standard classification systems is one of the strengths of our approach. Understanding which life-events play equivalent roles in human lives is one of the aspects which allow for improved classification and recommendation.

2.5 Person-summaries: Understanding the representation of individuals

Along with the concept representations described above, `life2vec` creates dense representations of individual life-sequences, *person-summaries*. The person-summary is a single vector that encapsulates the essential aspects of an individual's entire sequence of life-events; the person-summaries span our person embedding space. To form a person-summary, the model determines which aspects are relevant to the task at hand. In this sense, the person-summaries are conditioned on a specific prediction task. Below, we focus on person-summaries for the case of mortality likelihood, but person-summaries relative to, e.g., change in the area of residence or choice of the university would be drastically different.

Overview of the person-summaries. The space of person-summaries is visualized in Fig. 5 A-G. Relative to the mortality prediction, the model organizes individuals on a continuum from low to the high estimated probability of mortality (the point cloud in panel D). In Fig. 5, we show true deceased through purple diamonds, while the confidence of predictions [73] is demonstrated via the radius of points (e.g. dots with a small radius are low-confidence predictions). Further, the estimated probability is displayed using a color map from yellow to green. We zoom in on two regions: Region 1, which shows an area with a high probability of the 'survive' outcome, and Region 2, with a high probability of the 'death' outcome. We see that while Region 2 has a majority of elderly individuals, we still see a large fraction of younger individuals (Fig. 5 E) and that it contains a fraction of true targets (Fig. 5 F). Region B has a largely opposite structure, with a majority of young individuals but a substantial number of older individuals as well (Fig. 5 E) and only a single actual death (Fig. 5 F). When we look into actual deaths in the low probability region, we find that the five deaths nearest to and in Region 1 have the following causes – two accidents, malignant neoplasm of the brain (C71.9), malignant neoplasm of cervix uteri (C53.8), and myocardial infarction (I21.9),

all causes of death that we would expect to be difficult to predict from life-event sequences.

Directions in the person embedding space using TCAV. Topic Concept Activation Vectors (TCAV) [36], give us a way to understand the meaning of directions in the person embedding space using labeled data. The idea behind TCAV is to use binary labeled data (e.g., the labels 'employed'/'unemployed') and identify the hyperplane that best separates those labels. The vector orthogonal to this hyperplane gives us a direction for 'employed'-'unemployed' in the embedding space (the Concept Activation Vector [36]). We then use this employment-direction to understand how that label impacts decisions. Specifically, we measure how moving our decision boundary along this direction changes predictions; how the prediction reacts to these changes is called the *concept sensitivity*.

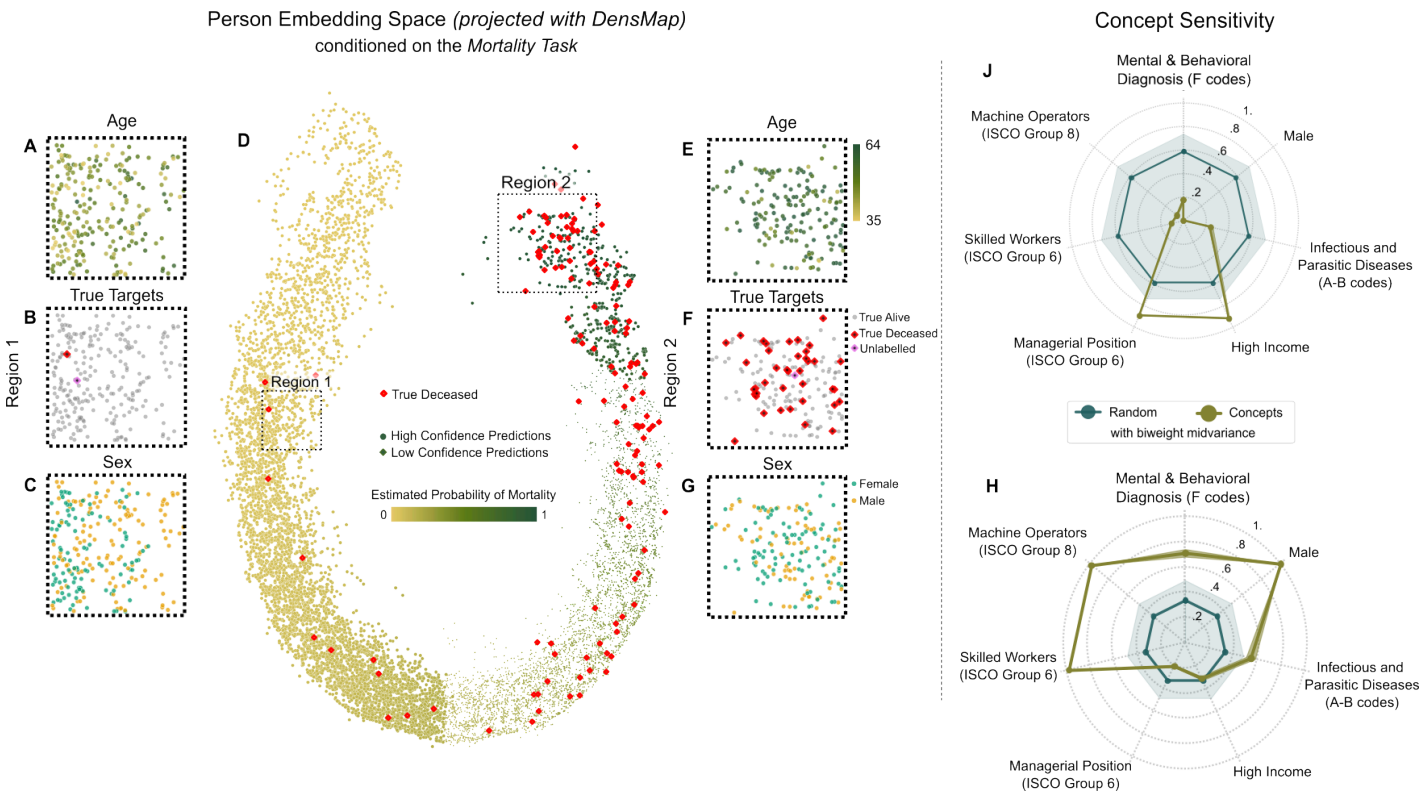

Figure 5: Representation of life-sequences conditioned on the Mortality Predictions. (**A-G**) Two-dimensional projection of 280-dimensional life representations(with the DensMap method [74]). (**D**) The full projection is colored based on the estimated probability of mortality. Pink points stand for the true deceased targets. Points with a smaller radius are uncertain predictions. (**A-C** and **E-G**) Zoomed-in regions with additional aspects associated with the life-sequence. (**A-C**) Region A contains points with a low probability of mortality, while (**E-G**) Region B contains points with a high probability. (**J-H**) Spider plot of `life2vec`'s concept sensitivity. The blue line is a median score for the random concept directions, while the blue area specifies the variation of the scores for the random concepts (**J**) Concept Sensitivity with respect to "Alive" prediction. (**H**) Concept sensitivity with respect to the "Deceased" prediction.

Fig. 5 J,H show concept sensitivity scores for several labels relative to the mortality-prediction

task. Here we show a two-dimensional projection using DensMap [74], but a range of other low-dimensional projections (T-SNE [75], UMAP [76], PaCMAP [72]) are available in SI (Sec.: Visualisation of Embedding Spaces). We focus on health-related labels such as a history of mental disease (or its absence), nervous system disease, diagnosis of neoplasm, and 'endocrine, nutritional and metabolic diseases.' Similarly, we use socio-economic attributes as labels – to measure the model's sensitivity to major occupational groups, sex, education, and origin. Fig. 5 J shows labels in relation to the prediction 'survive', and Fig. 5 H shows concepts with respect to the prediction 'death' within the four years following our sequence. Values close to one imply that moving in the topic direction indicates that moving in the label-direction increases the probability of a specific outcome. Values close to zero indicate the opposite. The gray areas are what we would expect if we moved in a random direction. We see that directions of possessing a managerial position or having a high income nudge the model towards the 'survive' decisions (Fig. 5 J), while being male, a skilled worker, or having a mental diagnosis has the opposite effect (Fig. 5 H). Note that while the spider plots in Fig. 5 J,H are almost mirrors, they are created based on different data sets, a further validation of robustness.

To confirm the validity of the sensitivity scores, we further perform extensive significance testing (Methods, Sec. 4.5). Our final approach to understanding the person-summaries is via inspection of the model's attention to individual sequences [35, 34, 77] – these confirm the findings discussed above (SI: Interpretability).

3 Discussion

Drawing on the progress from the natural language processing that made ChatGPT [78] possible and a massive nation-scale dataset that captures small and large events in the lives of millions of individuals over a decade, the `life2vec` model builds complex contextual representations of a range of aspects that characterize human lives: health, occupation, geography, and wealth.

When we draw on these representations to make predictions, transformer-based `life2vec` is able to adapt to different settings, from death-prediction to personality nuances, yielding highly accurate predictions that outperform state-of-the-art baselines trained on the same dataset.

When we investigate how the model can make these predictions, we find that to solve these diverse tasks, the model relies on different aspects of life trajectories. Mortality prediction requires the model to estimate how single events impact future outcomes while predicting personality nuances extracts information from large-scale patterns in the trajectories. More than that,`life2vec` handles the distinct complications of each task, such as missing labels,

imbalanced sample sizes, and ordinal multi-label settings.

We can shed further light on what the algorithm learns by studying its embedding spaces. The highly structured concept embedding space contains the model's fundamental building blocks. Here, we show that the model captures a meaningful and robust relationship between tokens of the vocabulary. Clusters emerge structured around concept tokens. Tokens tend to cluster according to classification systems (e.g., ICD-10, ISCO-08), revealing local relationships (how highly related tokens relate to one another) as well as global (how high-level concept-groups relate to one another) semantic relations in the system.

The model also captures the ordinal nature of features such as time, year, and income. Finally, the model converges to a similar embedding space given different subsets of data (and space is not biased with respect to frequent tokens).

In the person embedding space, the model produces representations that condense signals from the entire life-sequence into a single vector. These representations are always conditioned on specific prediction tasks. We can probe the person embedding space to gain intuition on why the model makes a certain prediction. Here, we find that in many cases model relies on relevant information (health, age, and income for the mortality prediction). However, we can also identify less obvious patterns, such as the role of the job-type. We can use the insights drawn from these summaries to generate new hypotheses and as a starting point for studies that focus on causality.

In summary, `life2vec` opens a range of possibilities within the social and health sciences. Through a rich dataset, we capture a wealth of complex patterns and trends in individual lives and represent their stories in a compact vector representation. These vectors represent a new type of comprehensive linkage between social and health outcomes. The output of our model, coupled with causality tools, shows a path to (a) systematically explore how different data modalities are correlated and interlinked and (b) use these interlinkages to explicitly explore how life impacts our health and vice versa. In this sense, we open the door to a new and more profound interplay between the social and health sciences. Finally, we stress that our work is an exploration of what is possible but should only be used in real-world applications under regulations that protect the rights of individuals (see Methods, Sec. 4.1).

4 Methods

4.1 Ethics and Broader Impacts

The data analysis was conducted at *Statistics Denmark*, the Danish National Statistical Institution. The data analysis was conducted under the Danish Data Protection Act and the General Data Protection Regulation (GDPR) [79]. In this context, since the data was used for scientific/statistical purposes, the usage is partially exempt from the GDPR [79] (e.g. from the right to be forgotten). Danish-based academic researchers, government agencies, NGOs, and private companies can be given access to Statistics Denmark data, but access is only granted under strict information security and data confidentiality policies[1] that ensure that data on individual entities are not leaked or used for purposes other than scientific/statistical. This focus on safekeeping data is shared with most other National Statistical Institutions that provide similar services. Using scientific/statistical 'products' such as `life2vec` for automated individual decision-making, profiling, or accessing individual-level data that may be memorized by the model is strictly disallowed. Aggregate statistics, including those coming from model predictions, may be used for research and to inform policy development.

We stress that `life2vec` is a research prototype, and in its current state, it is not meant to be deployed in any concrete real-world tasks. Before it could be used, e.g., to inform public policies in Denmark, it should be audited, in particular, to ensure the demographic fairness [80] of its predictions (with respect to the appropriate fairness metrics for the given context) and explainability [81] (e.g. if used for assisting decision-making based on synthetic/counterfactual data). Such audits would likely soon be mandated by the AI Act[2], focusing on the safe use of 'high-risk' models. Further auditing information is located in SI: Model Card.

Finally, we note that while it is possible that phenomena captured by `life2vec` reflect phenomena that have similar distributions outside of Denmark (e.g., labor market trajectories, individual health trajectories) – we urge caution with extrapolation to other populations since we have not explored how our findings translate beyond the current study population.

4.2 Dataset

We work with the Labour Market Account (`AMRUN`) [11] and the National Patient Registry (`LPR`) datasets [13, 40]. Within the Labour Market Account dataset are event data for every resident of Denmark. For Danish residents who have been in contact with secondary of health care services, primarily hospitals, the events are accounted in the National Patient Registry. We limit ourselves to data recorded in the period from 2008 until the end of 2015. Datasets are pseudonymized prior to our work by de-identifying addresses, Central Person Register numbers (CPRs), and names. Data is stored within Statistics Denmark, and all access/use of data is logged.

The total number of residents in the filtered dataset is *3 252 086*. For our research, we choose people who (1) are alive and lived in Denmark on the 31st December 2015, (2) have at least 12 records in the

[1] `https://www.dst.dk/en/OmDS/strategi-og-kvalitet/datasikkerhed-i-danmarks-statistik`
[2] `https://www.europarl.europa.eu/thinktank/en/document/EPRS_BRI(2021)698792`

labor data during the year of 2015[3], (3) have consistent sex and birthday attributes over the whole residency period, (4) are between 25 and 65 years old on the 31st December 2015.

These prerequisites apply for both stages: pre-training and finetuning (mortality prediction and self-reported personality questionnaires).

For the mortality prediction task, we excluded young individuals with very low death rates and older individuals with a high background probability of death. Thus, we narrowed the specification of requirements (4) and limited the dataset to people who are between 35-55 years old on 31st December 2015 (which limits us to *2 301 993* people).

For the personality nuances prediction task, we do not alter the initial requirement (4) but add new requirements on top of the original ones: (5) residents should have participated in the POSAP study [71], and (6) none of the scores associated with any HEXACO personality nuance (facet, dimension) are missing. This results in analyzing responses of *9 794* people.

Specifically, in POSAP HEXACO-60 [82] was administered, comprising 60 items (each representing one personality nuance) that can further be aggregated in (24) personality facets and, in turn, six personality dimensions (Honesty-Humility, Emotionality, Extraversion, Agreeableness vs. Anger, Conscientiousness, Openness to Experience). T

4.2.1 Labour Data

The Labour Marked Accounts dataset [11] contains data on each taxable income a resident receives, such as a salary, state scholarship, pension, etc. Each taxable income has multiple associated features, we focus on 16 features, see Tab. A4. Some of these features are linked to the workplace: *Type of Enterprise* [83], *Industry Code* [39]. Others describe personal attributes: *Professional Positions* [38], *Labour Force Status*, *Labour Force Status Modifier*, *Residential Municipality*, *Income*, *Working hours*, *Tax Bracket*, *Age*, *Country of Origin* and *Sex*.

Types of Enterprise feature is based on *European system of accounts (ESA2010)* [83], while Industry codes are encoded with Danish Industry Code (DB07) [39]. Industry codes provide information about the type of services the company offers. For example, code 108400 stands for the 'Preparation of flavorings and spices', and 643040 stands for the 'Venture companies and private equity funds'. ESA2010 has an intrinsic structure, which allows us to use more general categories (i.e., only the first four digits of a code).

Job types are classified via the *International Standard Classification of Occupations* (ISCO-08) [38]. The system encodes job types with four digits, e.g., code 2111 references 'physicists and astronomer', while code 5141 references 'barbers'. However, several codes exceed the length of 4, and since ISCO-08 also has hierarchies, we can collapse those to four-digit codes.

Labour Force Status provides information about a person's attachment to the Labour Market. The attachment does not solely include different forms of employment. For example, for a person enrolled in an official higher educational program, the status would be a 'student'. Being unemployed is also a

[3]Corresponds to 12 incomes over one year (e.g. salary, pension, etc.). We do not set requirements on the health-set as not every resident has any records in the health dataset

type of attachment, even though the financial compensation is not a salary. Some labor force statuses have additional information in the form of a modifier. If present, the modifier gives specifications for the labor force status. If the labor force status is student, the modifier might specify a 'foreign student'. A person can have multiple labor force statuses in the same period of time. Using the student example again, a student can also have employment alongside studying, and both would be accounted for in the dataset.

Since we want to have a concept token representation of continuous variables, such as income and labor-force-period, we binarize them based on quantiles. For example, the income variable is split into 100 categories. Another continuous variable is the labor-force-period. It is a percentage of days in a month that the Labour Force status is relevant for (binned in 10 categories). We also reserve concept tokens for each birth year and birth month.

4.2.2 Health Data

The health data pertains to all ambulatory and inpatient contacts with hospitals in Denmark. The country has a publicly funded healthcare system that caters to all citizens. The data is encoded using the ICD-10 System [40], an internationally authorized WHO system for classifying procedures and diseases. This system encompasses approximately 70,000 procedures and 69,000 diseases, each term represented by up to 7 symbols. The first symbol denotes the chapter, which represents a specific type of diagnosis. The first three symbols combined provide the category. For example, code `S86` is in chapter `S`, which stands for the 'injuries and poisoning' and `S86` combined stands for the 'injury of muscle, fascia, and tendon at lower leg level'. By adding or removing symbols, one can control the specificity of the term.

To reduce the vocabulary size, we collapsed all codes to the category level, which resulted in 704 terms. The data includes patient type, emergency status, and urgency in addition to diagnoses. Patient type denotes the admission type, i.e., inpatient, outpatient, or emergency. Emergency status indicates a patient admitted via an Emergency Care Unit, while urgency specifies whether the cause of admission was an acute onset.

4.2.3 Preprocessing

Each health and labor record is translated into a sentence, where each associated attribute (e.g., diagnosis, job type) is converted to a concept token. For example, if a labor record is connected to a job type 'Work with archiving and copying' (code `9210` in ISCO-08 [38]), we convert it to `POS_9210`. As a result, we have two types of sentences: *labor sentences* and *health sentences*. For each resident, we also create a *background sentence* that contains information about the birth month, birth year, country of origin (i.e., Denmark or Rest), and sex (SI: Specification of features and their sources)

4.2.4 Sentence and Document Structure

For each resident $r \in \{1, 2, 3, ..., R\}$ in the dataset $\mathcal{D}$, we assemble a chronological sequence of labor and health events. Each life-sequence has a form $S_r = \{s_r^0, s_r^1, s_r^2, ..., s_r^{n_r}\}$, where s_r^i is the i-th life-event

of the r-th resident.

Each event, s, contains tokens $v \in \mathcal{V}$ associated with a particular life-event, where $\mathcal{V}$ is a vocabulary of our artificial language. Along with the concept tokens, each event has associated temporal information such as absolute position, age, and segment. $\mathcal{P}$ is a set of possible absolute temporal positions, where p is the number of days passed between the event, s, and the *origin point* of 1st January 2008 (the day our dataset starts). If an event happened on the 24th of February 2012, then $p = 1516$. $\mathcal{A}$ is a set of possible age values: a specifies the number of *full* years passed since the person's birthday up until the date of the event, s. In terms of the `life2vec` model, p contextualizes events on a *global* (or universal) time scale, while a contextualizes events on the *individual* timeline.

Lastly, $\mathcal{G}$ is a set of segments. In case two or more events happen on the same day (and thus, share identical age and absolute position), segment information adds additional positional information. We have three distinct segments, and each life-event has an assigned segment value, g. The `life2vec` model learns the embedding of each segment.

The vocabulary set, $\mathcal{V}$, also includes several special tokens. For example, `[CLS]` starts a sequence and is later used to encapsulate a dense representation of the sequence. `[SEP]` token stands between the events, `[UNK]` substitutes concept tokens that are not in our vocabulary (e.g., tokens that were removed due to the low appearance frequency).

When we refer to the sentence length, $\|s\|$, we refer to the number of the corresponding concept token. The length of every sentence, s, varies depending on the type of the event it describes – health events range from two to three tokens, while labour-events range from three to seven concept tokens. Thus, the final length of the sequence, $\|S_r\|$, is a sum of the length of all the events, plus the number of special tokens such as `[CLS]` and `[SEP]`.

The first sentence in the sequence, s_r^0, is a *background sentence* that consists of gender, origin, birth-year, and birth-month tokens. It does not have associated age or absolute time position but does have segment information.

The maximum length of the document is 2560 concept tokens. If the length of the document, $\|S_r\|$, is above the specified limit, we remove earlier events (without removing a background sentence) until we can fit all the tokens of the last sentence (plus, last `[SEP]`). In case the length of the document is below the limit, we add padding tokens, `[PAD]`, at the end of the sequence to fill up the empty spaces.

4.2.5 Data Split

Finally, we randomly split the dataset (filtered according to (1), (2), (3), and (4) initial requirements) into training, validation, and test sets with a ratio of 70/15/15. The random split is *independent of any features* of the sequence (entirely at random). The global training set has 2 276 460 people, the global validation set has 487 812 people, and the global test set has 487 812 people. We preserve the splits for the finetuning tasks but remove records that do not satisfy specific requirements.

4.3 Model architecture

The model consists of three components: an embedding layer, a Bert-like encoder [31], and task-specific decoders. The encoder is a transformer-based model, while decoders are fully-connected neural networks.

4.3.1 Inputs and Embedding Component

The first step of the pipeline is to convert life-sequences into dense representations. Given a sequence S_p, we look up representations of tokens in the embedding matrix $\mathcal{E}_\mathcal{V} : \mathcal{V} \rightarrow \mathbb{R}^d$, where each row of $\mathcal{E}_\mathcal{V}$ corresponds to a token in the vocabulary (d is the number of hidden dimensions). Additionally, we look up the segment embedding in the $\mathcal{E}_\mathcal{G} : \mathcal{G} \rightarrow \mathbb{R}^d$ matrix. Both $\mathcal{E}_\mathcal{V}$ and $\mathcal{E}_\mathcal{G}$ matrices are optimized during the model training. To improve the representation of rare concept tokens and the overall isotropy of the concept embedding space [84], we remove the global mean from each row of the $\mathcal{E}_\mathcal{V}$ matrix [84]. That is, each time we look up the token embedding, we subtract the mean.

Regarding age and absolute time positions, we use the `Time2Vec` [54] method designed to model the linear and periodic progression of time. It introduces two learnable parameters: ω and φ. These determine the frequency and phase of periodic functions. The dense representations of age and position are calculated by the following equation, where z specifies the number of dimensions. We initialize two separate sets of time2vec parameters – one for the age, $\mathcal{T}_\mathcal{A} : \mathcal{A} \rightarrow \mathbb{R}^d$, and one for the absolute time position, $\mathcal{T}_\mathcal{P} : \mathcal{P} \rightarrow \mathbb{R}^d$. In both cases, we use the cosine function:

$$\mathcal{T}(x)[z] = \begin{cases} \omega_z x + \varphi_z & \text{, if } z = 0 \\ cos(\omega_z x + \varphi_z) & \text{, if } 1 \leq i \leq k. \end{cases}$$

The temporal representation of a sentence, s_r, is calculated according to Eq. 1. Scalars α, β, and γ are trainable parameters [44] initialized at a zero value.

$$\mathcal{E}_{temp}(s_r) = \alpha \cdot \mathcal{T}_\mathcal{A}(a) + \beta \cdot \mathcal{T}_\mathcal{P}(p) + \gamma \cdot \mathcal{E}_\mathcal{G}(g) \tag{1}$$

For each token v in s, we sum the associated token embedding in $\mathcal{E}_\mathcal{V}(v)$ and the temporal embedding of the sentence, $\mathcal{E}_{temp}(s_r^i)$. The input to the `life2vec` model is a concatenated sequence of these token representations.

4.3.2 Encoder Component

Like the original BERT [31], the `life2vec`-encoder consists of multiple encoder blocks. Each block processes input representations and passes the results to the next encoder. The architecture of each block is identical and consists of Multi-Head Attention, a Position-wise layer, and two residual connections (SI: Implementation Details).

The Multi-Head Attention module consists of several attention *heads*, which separately process the input representations. Vanilla BERT [31] uses softmax self-attention heads. Each head takes input

representations and transforms these with several dense layers - *query*, *key*, and *value*. These layers output linearly-transformed representations $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{L \times d}$, where L is the length of the sequence and d is the dimensionality of embeddings. The contextualised representations are computed as (Note that 1_L is a vector of ones with the length of L):

$$Att(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{d}})\mathbf{V} \Leftrightarrow \mathbf{D}^{-1}\mathbf{A}\mathbf{V}, \tag{2}$$

$$\text{where } \mathbf{A} = \exp(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{d}}),\ \mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1}_L)) \tag{3}$$

Softmax attention is unstable for sequences of length more than 512 tokens [85]. Therefore, we use softmax attention heads only to model local interactions, i.e., we limit the span of these heads to 38 neighboring tokens.

To capture global interactions, we use Performer-style attention heads [30], as they can handle longer sequences. Instead of calculating the precise attention matrix $\mathbf{A} \in \mathbb{R}^{L \times L}$, Performer-heads approximate it via matrix factorization. Entries of the approximated attention matrix are computed using kernels $\mathbf{A'}(i,j) = K(\mathbf{q}_i^T, \mathbf{k}_j^T)$ (indexes stand for the rows of matrices). The kernel function is defined as $K(\mathbf{x}, \mathbf{y}) = \mathbb{E}[\phi(\mathbf{x})^T, \phi(\mathbf{y})]$, where $\phi(\mathbf{u})$ is a random feature map that projects input into the r-dimensional space. Random mapping ϕ is constrained to contain features that are positive and exactly orthogonal (for details, refer to [30]). If we apply ϕ to $\mathbf{Q}, \mathbf{K}$, we get $\mathbf{Q'}, \mathbf{K'} \in \mathbb{R}^{L \times r}$, where $r \ll L$. The attention is now defined as:

$$\overline{Att}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \hat{\mathbf{D}}^{-1}(\mathbf{Q'}(\mathbf{K'}^T\mathbf{V})),\ \text{where } \hat{\mathbf{D}} = \text{diag}(\mathbf{Q'}(\mathbf{K'}\mathbf{1}_L)) \tag{4}$$

Each Multi-Head Attention module of the `life2vec` transformer has four Performer-style attention heads and four Softmax Attention Heads (SI: Attention Mechanism). The output of these heads is concatenated and transformed with one more dense layer.

The encoder blocks also have a Position-wise Feed-Forward module (`PFF`). It consists of two fully connected feed-forward layers that apply additional non-linear transformations to each representation: $f_{\text{PFF}}(\mathbf{x}) = \texttt{swish}(\mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\,\mathbf{W}_2 + \mathbf{b}_2$, where $\texttt{swish}(\mathbf{x}) = \mathbf{x} \cdot \texttt{sigmoid}(\mathbf{x})$ [45].

Typically, the output representations of each module add up to the input representations: $\mathbf{y} = \mathbf{x} + f(\mathbf{x})$ [31], where f is a Multi-Head Attention module or a Position-wise Feed-Forward module. In our work, we use `ReZero` connections [44], consisting of a single scalar, α. This scalar controls the fraction of information that each layer contributes to the contextualized representations: $\mathbf{y} = \mathbf{x} + \alpha \cdot f(\mathbf{x})$. At the start of training, each α is initialized to zero (meaning that none of the layers contribute. We introduced several modifications to BERT architecture, such as `ReZero` [44], `ScaleNorm` [46], `Swish` [45], and Weight Tying [47, 48] to speed up the convergence and reduce the size of the model.

4.4 Training procedure

The training procedure is split into two stages: learning the overall structure of the data (pre-training) and task-specific inference (finetuning).

4.4.1 Pre-training: Learning Structure of the Data

During the pre-training stage, `life2vec` learns embeddings of concept tokens and optimizes the parameters of the encoder component. The training objective consists of two tasks: Masked Language Modeling (MLM) [52] and Sequence Order Prediction (SOP).

The Masked Language Modeling task forces the model to learn relations between concept tokens. We randomly choose 30% of the tokens in the input sequence [86]. 80% of the chosen tokens are substituted with `[MASK]`, 10% are unchanged, and 10% are substituted with random tokens [31]. We do not mask any special tokens such as `[CLS]`, `[SEP]`, `[PAD]`, or `[UNK]` (nor do we use them as random tokens). We use altered sequences as inputs to `life2vec`. Using the contextual output representations of tokens, the model should infer the masked tokens.

The MLM decoder consists of two fully connected layers (f_1 and f_2). Each contextual representation, $\mathbf{x}_i$, is transformed via $f_1(\mathbf{x}) = \texttt{tanh}(\mathbf{x}\,\mathbf{W}_1 + \mathbf{b}_1)$, followed by l2-normalisation, $\texttt{norm}(\mathbf{x}) = \mathbf{x}/\|\mathbf{x}\|$. The weights of the final layer, f_2, are tied to the embedding matrix, $\mathcal{E}_\mathcal{V}$, which is further normalized to preserve only directions [48]. The resulting scores is scaled by α to sharpen the distribution [46]

$$\text{MLM}(\mathbf{x}) = \alpha \cdot f_2(\,\texttt{norm}(f_1(\mathbf{x})\,) \tag{5}$$

For each masked token the model must uncover, the decoder returns the likelihood distribution over the entire vocabulary. The likelihood (in our case) is a product of the scaled cosine distance between the contextualized representation of a token and the original representations of tokens in $\mathcal{E}_\mathcal{V}$ [48, 47].

The Sequence Order Prediction task forces the model to consider the progression of a life-sequence. It is an adapted version of the Next Sentence Prediction task [52]. Each life-event in the sequence has four attributes: concept tokens, segments, absolute time position, and age. In 10% of cases, we exchange concept tokens of one life-event with the concept tokens of another life-event (while preserving the positional and temporal information). In half of these cases, the exchange *reverses* the sequence so that 1st life-event exchanges tokens with the last life-event, the second life-2vent exchanges tokens with the second-to-last event, etc. In the other half, we *randomly* pick pairs of life-events to exchange the concept tokens.

The SOP decoder pulls the contextual representation of the `[CLS]` token from the last encoder layer and passes it through two feed-forward layers to make a final prediction

$$\text{SOP}(\mathbf{x}) = \texttt{ScaleNorm}\,[\,\texttt{swish}(\mathbf{x}\,\mathbf{W}_1 + \mathbf{b}_1)\,]\;\mathbf{W}_2 + \mathbf{b}_2 \tag{6}$$

4.4.2 Finetuning: Task Specific Training

On finetuning, we initialize the model with the optimized parameters from the pre-training stage and assign a new task to the model (i.e., remove the `MLM` and `SOP` encoders), which involves initializing a new decoder network.

We evaluate the `life2vec` model in two settings: Mortality Prediction and Personality Nuances Prediction. For the Mortality Prediction task, we pool the contextualized representation of each token in the sequence (i.e., the output of the last encoder layer) and use a weighted average of these tokens [87] to generate Sequence Representations. For the Personality Nuances Prediction Task, we only pool the

contextualized representation of the [CLS] token and pass it through a decoder network to make a prediction. The output of the decoder's second-to-last layer is also a Sequence Representation. Refer to SI: Model Architecture for more details.

The weights of the encoder model are updated during the finetuning. However, deeper encoders have a lower learning rate to avoid 'catastrophic forgetting' [88]. We also freeze the parameters of $\mathcal{E}_{\mathcal{V}}$, except for the parameters associated with [CLS], [SEP] and [UNK] tokens.

Mortality Prediction is a binary classification task. The goal is to infer the mortality likelihood within the next four years after 1st January 2016 (i.e., labels are *alive* and *deceased*).

The crucial aspect of the mortality prediction is the *loss function*. The data we use (see Sec. 4.2) includes people who might have left the country or disappeared before the end of 2020. Hence, we have a handful of *right-censored* outcomes. Using a Cross-Entropy loss would bias the predictions as we do not know the true outcome of all the sequences. Thus, we view the task as a Positive-Unlabeled Learning [59] problem. We assume that all negative samples and samples with missing labels make up the unlabeled set, while all positive samples make a Positive-labeled set (see SI: Implementation Details).

Personality Nuances Prediction Task is an ordinal classification task where labels correspond to the level of agreement with a particular item/statement (five levels). We predict the response to four different items simultaneously.

Predicting agreement levels poses two technical issues. First, responses are unevenly distributed across possible answers, with a majority choosing non-extreme answers, and second, the level of agreement has an ordinal nature.

We therefore slightly modify the training procedure. To prevent overfitting to the majority class, we employ instance difficulty-based re-sampling [89]– samples that are hard to predict would be subsampled with more frequently (SI: Sec. E.6). To account for the ordinal and imbalanced nature of the data, we combine three loss functions – class distance weighted cross-entropy [90], focal loss [91] with label smoothing penalty [92] (SI: Loss Functions), and use a modified softmax function [93]

4.4.3 Baseline Models

To evaluate the performance of life2vec on the mortality prediction task, we use six baseline model majority class prediction, random guess, mortality tables, logistic regression, feed-forward neural network, and recurrent neural network (RNN) [94, 95]. We perform a hyperparameter optimization similar to the one we have done for the life2vec model (SI: Implementation Details).

- **Logistic Regression** is a generalized linear regression model. We optimize it using Asymmetrical Cross-Entropy Loss [59] with the ridge penalty and stochastic gradient descent. As an input to the model, we use a counts vector, i.e. how many times each token appears in a sequence over a one-year interval.
- **Life Tables** is a logistic regression model that uses *only* age and sex as covariates,
- **Feed-forward network** uses the counts vector. It has a similar optimization setting as logistic regression. It has multiple feed-forward layers stacked over each other.

- **RNN model** uses the same input as the `life2vec` model and the same optimization settings. RNN model outputs the contextual representation of each token, which we then pass through a decoder network (identical to the one in the `life2vec`'s one).

These models work with the same data (i.e., batches of data are identical) and the same optimization settings.

For the Personality Nuances Prediction Task, we use a random guess and the RNN model. The `life2vec` model pools only the `[CLS]` representation from the decoder; however, with the RNN model, we pool all the contextual representations from the RNN (this way, we improve the performance of the RNN-based model).

4.4.4 Data Augmentation

To stabilize the performance of the `life2vec` model, we introduce several data augmentation strategies. It was an essential part of the training procedure and helped boost the performance of `life2vec` and baseline models. The augmentation techniques include subsampling sentences and tokens, adding noise to the temporal information, and masking the background sentence (SI: Implementation Details).

4.5 Interpretability

To provide the local interpretability, we use the Gradient-based Saliency score with L2-normalisation [35, 77, 34]. The saliency score highlights the sensitivity of the output with respect to each input token, i.e., the higher the sensitivity score, the more the output changes if we change the token representation (SI: Implementation Details).

TCAV. Gradient-based Saliency is unreliable when we want to see the global sensitivity of a model towards certain concepts (on a global scale). The person-summaries (provided by the `life2vec`) form a complex multidimensional space. Dimensions of this space do not necessarily have human-interpretable meaning. Thus, we use Testing with Concept Activation Vectors (TCAV) [36] to estimate the overall sensitivity.

We define a high-level concept as a subsample of life-sequences that share specific attributes (such as "individual has an F-diagnosis in the sequence"). We can take sequence representations of this subsample and train a linear classifier to discriminate between sequences in concept and random subsamples. The normal to the decision hyperplane is a concept direction. To calculate the TCAV scores, we rely on the following procedure [36] (SI: Implementation Details).

4.6 Evaluation of the Concept Space

While the structure of the Concept Space (Fig. 4) seem reasonable under manual inspection, we provide further statistical proof for the robustness of the embedding.

To demonstrate the **robustness of the concept space**, we used randomization tests [96]. Here we test

if the model preserves the distances between pairs of concept tokens given different dataset splits.

We trained three models with identical architecture. Each model had a different random initialization and was trained on a unique subset of the training data for ten epochs.

Further, we extracted the trained concept embeddings and calculated the cosine distances between each concept for each model separately (we refer to these matrices as $\mathcal{M}_1$, $\mathcal{M}_2$, and $\mathcal{M}_3$). We also obtained the distance matrices based on the randomly initialized embeddings and on the permuted version of $\mathcal{M}_1$ (referred to as $\mathcal{R}$ and $\mathcal{P}$, respectively).

To prove that our embedding spaces preserve structure/distances, we test whether two matrices are correlated. To perform the comparison, we use Randomisation Test described in [96]. For each pair of matrices, we permute columns and rows of the first matrix and calculate the correlation between permuted and the second matrix. We run the procedure 1 000 times. As a result, we get a distribution of correlation coefficients under the null hypothesis that there is no relationship between the two matrices. Suppose the correlation between the initial matrices is higher than the randomized one (falls above 95-th quantile of a distribution); in that case, we can indeed assume that the two are similar and, thus, distances between concepts are similar. To account for the multiple testings $(\mathcal{M}_1,\ \mathcal{M}_2)$, $(\mathcal{M}_1, \mathcal{M}_2)$, $(\mathcal{M}_2, \mathcal{M}_3)$, $(\mathcal{M}_1, \mathcal{R})$, $(\mathcal{M}_1, \mathcal{P})$ we use Benjamini–Hochberg procedure [97]. We reject the null hypothesis in the first three pairs with $p \approx 3\text{e}{-4}$ in all cases and accept the null hypothesis in case of the random-comparison case ($p \approx .76$) and permuted-comparison case ($p \approx .37$).

Our evaluation shows that the concept space converges to a similar space structure for each subset of a dataset.

Hubness of the Concept Space. The embedding spaces produced by ML models often degenerate due to the presence of the low-frequency tokens [98, 84]. The model places tokens along a similar direction, leading to less meaningful representations. The presence of hubs (tokens with an abnormal number of neighbors) is a proposed proxy for the degeneration of the embedding space [99] (aka anisotropy).

To identify hubs in the embedding matrix, $\mathcal{E}_{\mathcal{V}}$, we found the five closest neighbors of each node based on cosine similarity and used the resulting adjacency matrix to create a directed graph. Hubs can be identified by counting the incoming edges, which are the tokens with a large number of incoming edges. However, we did not find any hubs (i.e., nodes with an abnormally large number of incoming connections). The `[PAD]` token has the highest number of incoming connections (i.e., 49 links), `[CLS]` (40 links), `[SEP]` (39 links), followed by `[Female]` (25), `[Male]` (24) – the token with the most incoming edges is neighbor to less than 2% of tokens. Thus, we do not find proof of a degenerated concept space.

In summary, `life2vec` produces a meaningful and robust representation of the building blocks of our synthetic language.

4.6.1 Evaluation Metric for Task-Specific Settings

Since **Mortality Prediction Task** is a PU-Learning task, we cannot use standard metrics to evaluate the model without introducing a bias [62]. We evaluate models using the *Corrected* Matthews Correlation Coefficient, C-MCC (see [62] for details), as well as the Area-Under the Lift (AUL) [58]. We also

provide the corrected balanced accuracy score and corrected F1-score (SI: Evaluation Details).

We use AUL for the model optimization as suggested in [58]. i.e., early stopping. AUL can be interpreted as the "*probability of correctly ranking a random positive sample versus a random negative sample*" [100].

We use bootstrapping to estimate the confidence intervals for the corrected C-MCC score.

For the **Personality Nuances Prediction Task**, we use Cohens's Quadratic Kappa (CQK) score to terminate the training (when the score decreases on the validation set) [90]. We also use CQK to evaluate and compare models.

References

[1] Jose Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788, 2018.

[2] Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2021.

[3] Daria Grechishnikova. Transformer neural network for protein-specific de novo drug generation as a machine translation problem. *Scientific reports*, 11(1):1–13, 2021.

[4] Roshan Rao, Joshua Meier, Tom Sercu, Sergey Ovchinnikov, and Alexander Rives. Transformer protein language models are unsupervised structure learners. In *International Conference on Learning Representations*, 2020.

[5] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Noam Shazeer, Curtis Hawthorne, Andrew M Dai, Matthew D Hoffman, and Douglas Eck. Music transformer: Generating music with long-term structure. *arXiv preprint arXiv:1809.04281*, 2018.

[6] Yi Zou, Pei Zou, Yi Zhao, Kaixiang Zhang, Ran Zhang, and Xiaorui Wang. Melons: generating melody with long-term structure using transformers and structure graph. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 191–195. IEEE, 2022.

[7] Yikuan Li, Shishir Rao, Jose Roberto Ayala Solares, Abdelaali Hassaine, Rema Ramakrishnan, Dexter Canoy, Yajie Zhu, Kazem Rahimi, and Gholamreza Salimi-Khorshidi. Behrt: transformer for electronic health records. *Scientific reports*, 10(1):1–12, 2020.

[8] Ashesh Chattopadhyay, Mustafa Mustafa, Pedram Hassanzadeh, Eviatar Bach, and Karthik Kashinath. Towards physically consistent data-driven weather forecasting: Integrating data assimilation with equivariance-preserving deep spatial transformers. *arXiv preprint arXiv:2103.09360*, 2021.

[9] Alabi Bojesomo, Hasan Al-Marzouqi, and Panos Liatsis. Spatiotemporal vision transformer for short time weather forecasting. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 5741–5746, 2021.

[10] Keyon Vafa, Emil Palikot, Tianyu Du, Ayush Kanodia, Susan Athey, and David M Blei. Learning transferrable representations of career trajectories for economic prediction. *arXiv preprint arXiv:2202.08370*, 2022.

[11] Danmarks Statistik. Arbejdsmarkedsregnskab. 2022.

[12] Ojvind Lidegaard, Christina H Vestergaard, and Mette Schou Hammerum. Kvalitetsmonitorering ud fra data i landspatientregisteret. *Ugeskrift for Laeger*, 171(6):412–5, February 2009.

[13] Elsebeth Lynge, Jakob Lynge Sandegaard, and Matejka Rebolj. The danish national patient register. *Scandinavian journal of public health*, 39:30–33, 2011.

[14] Carsten Bøcker Pedersen. The danish civil registration system. *Scandinavian journal of public health*, 39:22–25, 2011.

[15] Laura A Mansfield, Peer J Nowack, Matt Kasoar, Richard G Everitt, William J Collins, and Apostolos Voulgarakis. Predicting global patterns of long-term climate change from short-term simulations using machine learning. *npj Climate and Atmospheric Science*, 3(1):44, 2020.

[16] Yasminah Alali, Fouzi Harrou, and Ying Sun. A proficient approach to forecast covid-19 spread via optimized dynamic machine learning models. *Scientific Reports*, 12(1):1–20, 2022.

[17] Shoshana Zuboff. *The age of surveillance capitalism: The fight for a human future at the new frontier of power*. Profile books, 2019.

[18] Max Weber. *The theory of social and economic organization*. Simon and Schuster, 2009.

[19] Matthew J Salganik, Ian Lundberg, Alexander T Kindel, Caitlin E Ahearn, Khaled Al-Ghoneim, Abdullah Almaatouq, Drew M Altschul, Jennie E Brand, Nicole Bohme Carnegie, Ryan James Compton, et al. Measuring the predictability of life outcomes with a scientific mass collaboration. *Proceedings of the National Academy of Sciences*, 117(15):8398–8403, 2020.

[20] Matthew J Salganik. *Bit by bit: Social research in the digital age*. Princeton University Press, 2019.

[21] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[22] Justin Grimmer, Margaret E Roberts, and Brandon M Stewart. *Text as data: A new framework for machine learning and the social sciences*. Princeton University Press, 2022.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[25] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[26] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[27] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[28] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[29] Tian Cai, Hansaim Lim, Kyra Alyssa Abbu, Yue Qiu, Ruth Nussinov, and Lei Xie. Msa-regularized protein sequence transformer toward predicting genome-wide chemical-protein interactions: Application to gpcrome deorphanization. *Journal of chemical information and modeling*, 61(4):1570–1582, 2021.

[30] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

[31] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[32] Austin C Kozlowski, Matt Taddy, and James A Evans. The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review*, 84(5):905–949, 2019.

[33] Mohammad Taher Pilehvar and Jose Camacho-Collados. Embeddings in natural language processing: Theory and advances in vector representations of meaning. *Synthesis Lectures on Human Language Technologies*, 13(4):1–175, 2020.

[34] Shuoyang Ding, Hainan Xu, and Philipp Koehn. Saliency-driven word alignment interpretation for neural machine translation. *arXiv preprint arXiv:1906.10282*, 2019.

[35] Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. A diagnostic study of explainability techniques for text classification. *arXiv preprint arXiv:2009.13295*, 2020.

[36] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

[37] Adriano Lucieri, Muhammad Naseer Bajwa, Stephan Alexander Braun, Muhammad Imran Malik, Andreas Dengel, and Sheraz Ahmed. On interpretability of deep learning based skin lesion classifiers using concept activation vectors. In *2020 international joint conference on neural networks (IJCNN)*, pages 1–10. IEEE, 2020.

[38] ILO. *International Standard Classification of Occupations: ISCO-08*. International Labour Office, 2012.

[39] Danmarks Statistik. *Dansk Branchekode 2007: DB07 (Danish Industrial Classification of All Economic Activities 2007)*. Danmarks Statistik, v3 edition, 2015.

[40] World Health Organization et al. Icd-10: International statistical classification of diseases and related health problems (10th revision), geneva: World health organization. *PEOPLE WITH LEARNING DISABILITIES*, 341, 1992.

[41] Pranjul Yadav, Michael Steinbach, Vipin Kumar, and Gyorgy Simon. Mining electronic health records (ehrs) a survey. *ACM Computing Surveys (CSUR)*, 50(6):1–40, 2018.

[42] Zhongyang Han, Jun Zhao, Henry Leung, King Fai Ma, and Wei Wang. A review of deep learning models for time series prediction. *IEEE Sensors Journal*, 21(6):7833–7848, 2019.

[43] Arturo Moncada-Torres, Marissa C van Maaren, Mathijs P Hendriks, Sabine Siesling, and Gijs Geleijnse. Explainable machine learning can outperform cox regression predictions and provide insights in breast cancer survival. *Scientific reports*, 11(1):6968, 2021.

[44] Thomas Bachlechner, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W Cottrell, and Julian McAuley. Rezero is all you need: Fast convergence at large depth. *arXiv preprint arXiv:2003.04887*, 2020.

[45] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[46] Toan Q Nguyen and Julian Salazar. Transformers without tears: Improving the normalization of self-attention. *arXiv preprint arXiv:1910.05895*, 2019.

[47] Nikolaos Pappas, Lesly Miculicich Werlen, and James Henderson. Beyond weight tying: Learning joint input-output embeddings for neural machine translation. *arXiv preprint arXiv:1808.10681*, 2018.

[48] Nikolaos Pappas, Lesly Miculicich Werlen, and James Henderson. Beyond weight tying: Learning joint input-output embeddings for neural machine translation. *arXiv preprint arXiv:1808.10681*, 2018.

[49] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial intelligence and machine learning for multi-domain operations applications*, volume 11006, pages 369–386. SPIE, 2019.

[50] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[51] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[53] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3694–3702, 2021.

[54] Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus Brubaker. Time2vec: Learning a vector representation of time. *arXiv preprint arXiv:1907.05321*, 2019.

[55] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[56] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[57] Amin Naemi, Thomas Schmidt, Marjan Mansourvar, Mohammad Naghavi-Behzad, Ali Ebrahimi, and Uffe Kock Wiil. Machine learning techniques for mortality prediction in emergency departments: a systematic review. *BMJ open*, 11(11):e052663, 2021.

[58] Liwei Jiang, Dan Li, Qisheng Wang, Shuai Wang, and Songtao Wang. Improving positive unlabeled learning: Practical aul estimation and new training method for extremely imbalanced data sets. *arXiv preprint arXiv:2004.09820*, 2020.

[59] Cong Wang, Jian Pu, Zhi Xu, and Junping Zhang. Asymmetric loss for positive-unlabeled learning. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021.

[60] Anne Vinkel Hansen, Laust Hvas Mortensen, Claus Thorn Ekstrøm, Stella Trompet, and Rudi Westendorp. Predicting mortality and visualizing health care spending by predicted mortality in danes over age 65. *Scientific Reports*, 13(1):1–7, 2023.

[61] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):1–13, 2020.

[62] Rashika Ramola, Shantanu Jain, and Predrag Radivojac. Estimating classification accuracy in positive-unlabeled learning: characterization and correction strategies. In *BIOCOMPUTING 2019: Proceedings of the Pacific Symposium*, pages 124–135. World Scientific, 2018.

[63] Brent W Roberts, Nathan R Kuncel, Rebecca Shiner, Avshalom Caspi, and Lewis R Goldberg. The power of personality: The comparative validity of personality traits, socioeconomic status, and cognitive ability for predicting important life outcomes. *Perspectives on Psychological science*, 2(4):313–345, 2007.

[64] Robert R McCrae and Paul T Costa Jr. The five-factor theory of personality. 2008.

[65] Ingo Zettler, Isabel Thielmann, Benjamin E Hilbig, and Morten Moshagen. The nomological net of the hexaco model of personality: A large-scale meta-analytic investigation. *Perspectives on Psychological Science*, 15(3):723–760, 2020.

[66] Hans J Eysenck. Superfactors p, e and n in a comprehensive factor space. *Multivariate Behavioral Research*, 13(4):475–481, 1978.

[67] C.G. Jung. Psychological types or the psychology of individuation. 1923.

[68] René Mõttus, Timothy C Bates, David M Condon, Daniel K Mroczek, and William R Revelle. Leveraging a more nuanced view of personality: Narrow characteristics predict and explain variance in life outcomes. 2022.

[69] Anne Seeboth and René Mõttus. Successful explanations start with accurate descriptions: Questionnaire items as personality markers for more accurate predictions. *European Journal of Personality*, 32(3):186–201, 2018.

[70] Ross David Stewart, René Mõttus, Anne Seeboth, Christopher John Soto, and Wendy Johnson. The finer details? the predictability of life outcomes from big five domains, facets, and nuances. *Journal of personality*, 90(2):167–182, 2022.

[71] Det danske personligheds og sociale adfærdspanel. https://copsy.dk/posap/. Accessed: 2023-03-21.

[72] Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. Understanding how dimension reduction tools work: an empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *The Journal of Machine Learning Research*, 22(1):9129–9201, 2021.

[73] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

[74] Ashwin Narayan, Bonnie Berger, and Hyunghoon Cho. Density-preserving data visualization unveils dynamic patterns of single-cell transcriptomic variability. *bioRxiv*, pages 2020–05, 2020.

[75] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[76] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[77] Jasmijn Bastings and Katja Filippova. The elephant in the interpretability room: Why use attention as explanation when we have saliency methods? *arXiv preprint arXiv:2010.05607*, 2020.

[78] OpenAI. Gpt-4 technical report, 2023.

[79] European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation).

[80] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6), jul 2021.

[81] Nadia Burkart and Marco F Huber. A survey on the explainability of supervised machine learning. *Journal of Artificial Intelligence Research*, 70:245–317, 2021.

[82] MC Ashton and K Lee. A short measure of the major dimension of personality. *European Journal of Psychological Assessment*, 91(4):340–345, 2009.

[83] E Eurostat. European system of accounts esa 2010. *Official Journal of the European Union*, 174:56, 2013.

[84] Daniel Biś, Maksim Podkorytov, and Xiuwen Liu. Too much in common: Shifting of embeddings in transformer language models and its implications. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5117–5130, 2021.

[85] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[86] Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. Should you mask 15% in masked language modeling? *arXiv preprint arXiv:2202.08005*, 2022.

[87] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[88] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*, pages 194–206. Springer, 2019.

[89] Sihao Yu, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Zizhen Wang, and Xueqi Cheng. A re-balancing strategy for class-imbalanced classification based on instance difficulty. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 70–79, 2022.

[90] Gorkem Polat, Ilkay Ergenc, Haluk Tarik Kani, Yesim Ozen Alahdab, Ozlen Atug, and Alptekin Temizel. Class distance weighted cross-entropy loss for ulcerative colitis severity estimation. In *Medical Image Understanding and Analysis: 26th Annual Conference, MIUA 2022, Cambridge, UK, July 27–29, 2022, Proceedings*, pages 157–171. Springer, 2022.

[91] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[92] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[93] Sekitoshi Kanai, Yasuhiro Fujiwara, Yuki Yamanaka, and Shuichi Adachi. Sigsoftmax: Reanalysis of the softmax bottleneck. *Advances in Neural Information Processing Systems*, 31, 2018.

[94] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[95] Kirill F. Andreev, Dmitri A. Jdanov, Carl Boe, Michael Bubenheim, Dimiter Philipov, Vladimir M. Shkolnikov, and Pierre J Vachon. Methods protocol for the human mortality database. 2002.

[96] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:4, 2008.

[97] David Thissen, Lynne Steinberg, and Daniel Kuang. Quick and easy implementation of the benjamini-hochberg procedure for controlling the false positive rate in multiple comparisons. *Journal of educational and behavioral statistics*, 27(1):77–83, 2002.

[98] Yuxin Liang, Rui Cao, Jie Zheng, Jie Ren, and Ling Gao. Learning to remove: Towards isotropic pre-trained bert embedding. In *International Conference on Artificial Neural Networks*, pages 448–459. Springer, 2021.

[99] Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. *arXiv preprint arXiv:1702.01417*, 2017.

[100] Shangchuan Huang, Songtao Wang, Dan Li, and Liwei Jiang. Aul is a better optimization metric in pu learning. 2020.

[101] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[102] Rafael Müller, Simon Kornblith, and Geoffrey E Hinton. When does label smoothing help? *Advances in neural information processing systems*, 32, 2019.

A Definitions

Table A1: Table of Notations

$\texttt{PFF}$	$\triangleq$	Position-wise feed forward module
$\texttt{MLM}$	$\triangleq$	Masked Language Model
$\texttt{SOP}$	$\triangleq$	Sequence Order Prediction
S_r	$\triangleq$	a life-sequence of a r-th resident
s_r^i	$\triangleq$	i-th event in a life-sequence of a resident r
L	$\triangleq$	maximum length of a sequence
d	$\triangleq$	number of hidden dimensions
$\mathcal{E}_{\mathcal{V}}$	$\triangleq$	embedding matrix of concepts
$\mathcal{E}_{\mathcal{G}}$	$\triangleq$	embedding matrix of segments
$\mathcal{T}_{\mathcal{A}}$	$\triangleq$	time2vec embedding of the age
$\mathcal{T}_{\mathcal{P}}$	$\triangleq$	time2vec embedding of the absolute position
$\mathbf{ff}_i$	$\triangleq$	i-th fully connected layer
$\mathbf{W}_i$	$\triangleq$	weight matrix of the i-th layer
$\mathbf{b}_i$	$\triangleq$	bias vector of the i-th layer
g	$\triangleq$	trainable parameter
$\mathbf{A}$	$\triangleq$	Attention score matrix
$\mathbf{1}_L$	$\triangleq$	vector of $\mathbf{1}$s with the length of L
		<u>Functions</u>
$\texttt{Norm}(x)$	$=$	$\frac{x}{\lVert x \rVert}$
$\texttt{ScaleNorm}(x)$	$=$	$g \cdot \frac{x}{\lVert x \rVert}$ [46]
$\texttt{sigmoid}(x)$	$=$	$\frac{1}{1+e^{-1}}$
$\texttt{swish}(x)$	$=$	$x \cdot \texttt{sigmoid}(x)$ [45]
$\texttt{SigSoftmax}(x_i)$	$=$	$\frac{\texttt{exp}(x_i) \cdot \texttt{sigmoid}(x_i)}{\sum \texttt{exp}(x_j) \cdot \texttt{sigmoid}(x_j)}$ [93]

B Evaluation Details

Table A2: Corrected Matthew's Correlation Coefficient (MCC) and Area under the Lift (AUL) on the Mortality Prediction Task. 95%-Confidence intervals for the MCC based on the stratified bootstrapping. In both cases, the higher value is preferred. Model Size specifies the number of trainable parameters, we performed a hyperparameter tuning on RNN, FFNN, and Logistic Regression.

Model	MCC, 95%-CI	AUL	Accuracy, 95%-CI	F1-Score, 95%-CI	Model Size
L2V	**0.413 [0.410, 0.422]**	**0.845**	0.788 [0.782, 0.794]	0.443 [0.435, 0.451]	8.4m
RNN-GRU	0.369 [0.361, 0.378]	0.834	0.778 [0.771, 0.783]	0.395 [0.389, 0.402]	1.5m
FFNN	0.340 [0.332, 0.348]	0.822	0.768 [0.762, 0.774]	0.345 [0.339, 0.350]	8.4m
Logistic Reg	0.149 [0.142, 0.155]	0.735	0.639 [0.633, 0.645]	0.201 [0.198, 0.204]	2.0k
Life Tables	0.059 [0.051, 0.066]	0.650	0.555 [0.548, 0.562]	0.161 [0.158, 0.164]	3
Random	-0.005 [-0.011, 0.002]	0.497	0.496 [0.489, 0.503]	0.132 [0.128, 0.135]	-
Majority Class	0.0	0.497	0.5	-	-

Table A3: Corrected Matthew's Correlation Coefficient (MCC) and Area under the Lift (AUL) on the *Emigration* Prediction Task. 95%-Confidence intervals for the MCC based on the stratified bootstrapping. In both cases, the higher value is preferred.

Model	MCC, 95%-CI	AUL	Accuracy, 95%-CI	F1-Score, 95%-CI	Model Size
L2V	**0.168 [0.159, 0.177]**	**0.802**	0.731 [0.719, 0.744]	0.130 [0.125, 0.134]	8.4m
RNN-GRU	0.144 [0.136, 0.151]	0.786	0.714 [0.702, 0.726]	0.106 [0.103, 0.110]	1.5m
Random	0.000 [-0.001, 0.009]	0.504	0.499 [0.486, 0.513]	0.052 [0.049, 0.055]	-
Majority Class	0.0	0.504	0.5	-	-

C Specification of features and their sources

Table A4: Specification of features and their sources.

Type	Feature	Source	# Categories	Encoding
Background Information	Sex	KOEN	2 binary	Male, Female
	Birth Month	FOED_DAG	12	Jan-Feb
	Birth Year	FOED_DAG	45	1946-1991
	Country of Origin	OPR_LAND	2 binary	National or International
	Municipality of Residence	BOPAEL_KOM_KODE	97	Danish municipality codes
	Tax Bracket	ATP_BIDRAG_SATS_KODE	6	based on DST definitions
	Income Level	BREDT_LOEN_BELOEB	100	Quantile-based
Labour Records	Labour Force Status	SOC_STATUS_KODE	35	based on DST definitions
	Labour Force Status (Modification)	TILSTAND_KODE_AMR	58	based on DST definitions
	Labour-Force-Interval	TILSTAND_LAENGDE_ARR	10	Quantile based
	Industry Area (Company)	ARB_HOVED_BRA_DB07	290	Danish Industry Classification System
	Job type	DISCO_KODE	359	International Standard Classification of Occupations
	Enterprise Type (Company)	ARB_SEKTORKODE	15	European System of Accounts
Health Records	Diagnosis	C_ADIAG	704	ICD-10
	Urgency	C_INDM	3	Urgent, Non-Urgent, Emergency
	Patient Type	C_PATTYPE	2	In-, out- patient

D Hyperparameter Optimization

Table A5: Hyperparameter optimisation for the life2vec model. We use Bayesian search to find the optimal configuration of the parameters. The overall perplexity is calculated as a weighted sum of perplexities generated by the MLM task (0.7) and Sequence Order Prediction task (0.3). We train each model for 5 epochs and then pick 6 models with the lowest scores. Lastly, we train these six models for 30 epochs and choose the one with the lowest perplexity score on the validation set. Model Nr. 3 is the final configuration of the life2vec model.

ID	Overall Perplexity	Hidden Size	# encoders	# heads	# local heads	FF Hidden Size	# random features	Local Window Size
0	1.870	332	13	4	3	996	242	52
1	1.843	238	13	14	4	1586	263	32
2	1.859	208	13	8	3	2235	413	93
3	***1.835***	***280***	***5***	***10***	***7***	***2210***	***436***	***38***
4	1.925	80	11	10	6	1355	153	40
5	1.908	96	12	8	1	1790	360	41
6	1.881	184	6	4	1	709	469	19
7	1.857	196	5	14	11	1124	326	74
8	1.838	228	14	6	1	1605	490	65
9	1.859	208	12	4	1	2135	77	114
10	1.846	210	8	14	5	1615	356	49
11	1.889	312	8	13	12	1991	138	165
12	1.867	216	10	4	2	1839	223	99
13	1.886	154	12	14	1	1532	97	102
14	1.916	70	8	14	5	2074	410	41
15	**1.829**	**270**	**6**	**10**	**1**	**1964**	**275**	**99**
16	1.848	242	4	11	10	2386	120	80
17	1.884	168	7	12	7	1702	66	229
18	1.865	336	4	12	10	2432	512	14
19	1.848	336	4	8	4	2560	512	4
20	1.889	162	6	9	4	1214	188	137
21	1.886	220	9	11	8	2482	271	188
22	1.846	336	7	12	4	2560	512	68
23	1.873	336	10	8	7	2560	512	4
24	1.878	310	5	5	2	2049	241	108
25	1.867	322	6	14	9	882	282	150
26	2.036	270	9	10	0	2322	192	208
27	1.857	288	7	6	1	1964	325	154
28	2.014	144	9	6	0	1906	395	198
29	1.867	242	10	11	1	2285	158	140
30	1.925	301	8	7	5	1380	296	183
31	1.944	120	5	12	5	2391	361	130
32	1.851	252	7	9	1	1771	209	254
33	1.878	294	10	7	5	2506	316	167
34	1.873	300	5	10	9	2124	429	82
35	1.857	275	6	11	10	2019	452	118
36	1.946	261	4	9	5	1304	174	230
37	1.878	312	5	13	7	1713	376	169
38	**1.827**	**253**	**6**	**11**	**9**	**2439**	**133**	**120**
39	1.916	171	5	9	3	2168	297	254
40	1.857	273	7	7	3	1904	250	89
41	**1.835**	**286**	**7**	**13**	**9**	**1446**	**207**	**138**
42	1.859	234	4	9	4	2352	315	25
43	1.838	286	8	11	6	1213	346	55
44	1.848	260	6	13	12	556	114	116
45	1.881	238	9	14	10	2560	106	133
46	2.039	294	4	14	0	2560	236	53
47	2.031	336	6	14	0	1912	180	256
48	1.870	108	13	12	10	1388	98	103
49	1.952	70	7	10	10	1454	334	4
50	1.938	70	4	14	12	1096	497	4
51	1.933	66	4	3	2	877	231	256
52	1.838	294	13	14	8	2412	91	4
53	**1.829**	**336**	**14**	**14**	**2**	**1785**	**105**	**4**
54	1.862	252	10	14	2	2312	116	4
55	**1.832**	**336**	**14**	**14**	**2**	**2096**	**64**	**256**

Table A6: Hyperparameter optimization for the `RNN-GRU` (Mortality Prediction). We use Bayesian search to find the optimal configuration of the parameters. We optimize parameters that were the most sensitive with respect to the performance of the model (determined manually). We pick the model with the highest AUL score (on the validation set after 5 epochs). Model 18 has the most optimal configuration of the hyperparameters.

ID	Hidden Size	# layers	Dropout, %	Bidirectional	AUL
1	370	1	0.16	False	0.7898
2	292	7	0.49	False	0.7824
3	155	7	0.02	False	0.7902
4	137	5	0.39	False	0.7898
5	525	5	0.01	False	0.7903
6	361	3	0.48	False	0.7865
7	646	3	0.35	False	0.7865
8	372	2	0.26	True	0.7902
9	260	7	0.09	False	0.7897
10	408	1	0.48	True	0.7890
11	367	2	0.37	False	0.7890
12	415	1	0.37	True	0.7897
13	267	1	0.40	True	0.7910
14	64	4	0.28	True	0.7910
15	64	1	0.50	False	0.7873
16	304	4	0.00	True	0.7893
17	768	8	0.00	True	0.7888
18	**256**	**3**	**0.27**	**True**	**0.7912**
19	710	2	0.04	False	0.7888
20	490	8	0.26	False	0.7375
21	268	8	0.21	False	0.7902
22	657	8	0.18	True	0.7728
23	248	8	0.14	False	0.7882
24	520	1	0.10	True	0.7893

Table A7: Hyperparameter optimization for the Feedforward neural network (Mortality Prediction). We use Bayesian search to find the optimal configuration of the parameters. We optimize parameters that were the most sensitive with respect to the performance of the model (determined manually). We pick the model with the highest AUL score (on the validation set after 5 epochs). Model 7 has the most optimal configuration of the hyperparameters.

ID	Hidden Size	Layers	Dropout	LR	AUL
1	370	1	0.16	0.00039	0.7260
2	292	7	0.49	0.00043	0.6950
3	155	7	0.02	0.00378	0.7230
4	137	5	0.39	0.00374	0.7199
5	525	5	0.01	0.00303	0.7234
6	361	3	0.48	0.00431	0.7196
7	**646**	**3**	**0.35**	**0.00017**	**0.7290**
8	372	2	0.26	0.00503	0.7216
9	260	7	0.09	0.00296	0.7203
10	408	1	0.48	0.00595	0.7223
11	64	5	0.19	0.01000	0.7193
12	64	1	0.39	0.01000	0.7239
13	768	3	0.25	0.00001	0.6914
14	768	2	0.00	0.00171	0.7250
15	768	3	0.00	0.00109	0.7258
16	599	7	0.00	0.00718	0.7196
17	665	1	0.47	0.00874	0.7171
18	768	2	0.24	0.00041	0.7246
19	363	3	0.35	0.00736	0.7185
20	462	2	0.24	0.00065	0.7255
21	110	2	0.21	0.00700	0.7232
22	592	3	0.38	0.00866	0.7190
23	511	2	0.20	0.00273	0.7236
24	155	2	0.21	0.00244	0.7250

E Implementation Details

E.1 Ecnoder-Decoder Architecture

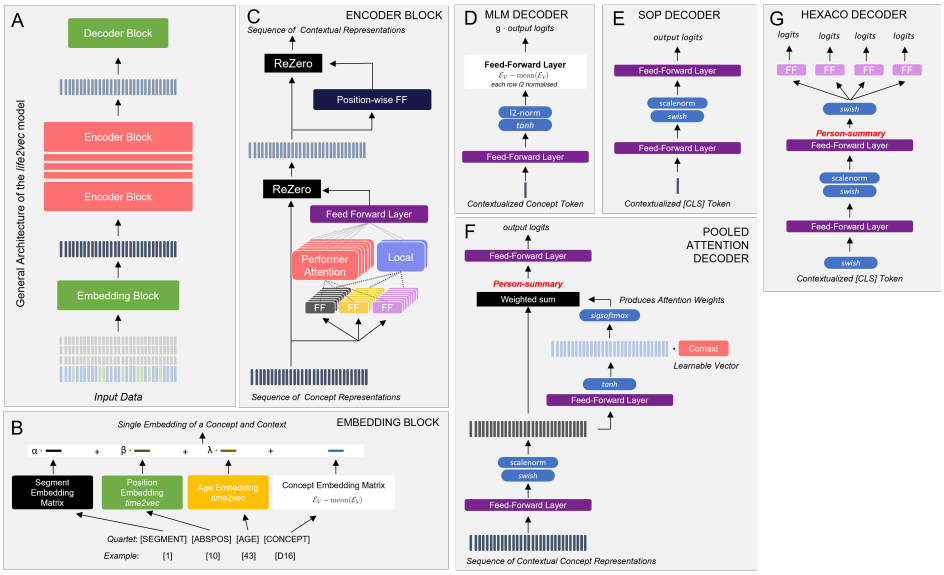

Figure A.1: Architecture of `life2vec`. (**A**) The overall structure of the model. The life-sequence is passed to the embedding block and every encoder block. The implementation of a decoder block depends on the task. Each block before the decoder outputs a contextualized representation of concept tokens. (**B**) Each concept token in a sequence (together with the positional information) passes through the embedding block. It merges positional and concept information. Age and Absolute Position embeddings are calculated via `time2vec` [54], while segment and concept embeddings are stored in a lookup matrix. To retrieve the concept embedding, we lookup the corresponding embedding in a matrix $\mathcal{E}_\mathcal{V}$ and then remove the mean of $\mathcal{E}_\mathcal{V}$ (without changing the representation in an actual $\mathcal{E}_\mathcal{V}$. Positional representations are weighted and added to the representation of a concept. (**C**) The encoder block takes the life-sequence representation from the previous block. The sequence is passed through the attention layer (Multi-head Performer and Local Attention). The result of the attention layer is added to life-sequence representation via `ReZero` gate. Further, the life-sequence representation is passed through the Position-Wise Feed Forward Layer. The result is added back to the life-sequence representation via another `ReZero` gate. (**D**) Masked Language Model Decoder pools (separately) contextual representation of masked concept tokens. MLM Decoder and the Sequence Order Prediction Decoder (**E**) are used during the pre-training. The last feed-forward layer copies weights from the matrix $\mathcal{E}_\mathcal{V}$. We remove the mean of $\mathcal{E}_\mathcal{V}$ from each row and then apply the l2-norm. Thus, the output logits are calculated as a dot product between the contextual representation of a concept and each row of standardized $\mathcal{E}_\mathcal{V}$ matrix [46]. (**E**) The Sequence Order Prediction decoder pools the contextual representation of `[CLS]` concept (which is always placed first in the sequence). (**F**) The pooled attention decoder is used during the finetuning. It pools the life-sequence from the last encoder block. After passing through the first feed-forward layer, it uses life-sequence representations to compute attention weights (right-hand side). It uses context (a trainable vector) to calculate the importance of a concept at a position i. Attention weights are then used to calculate the weighted average of concepts in the life-sequence, or *person-summary*. (**G**) HEXACO Decoder pools the contextual representation of `[CLS]` and computes the logits per each personality item.

E.2 Data Augmentation

We use data augmentation during the pre-training and fine-tuning stages. These techniques ensure robust performance and better generalizability of the model.

Sequence Downsampling. We randomly pick a life-sequence and randomly remove up to 50% of life-events.

Temporal Noise As labor-events mainly occur on the last day of a month, we smooth the distribution of the absolute time. We randomly pick a life-sequence and alter the absolute time by injecting noise, $\mathcal{U}(-5, 5)$, into each life-event.

Background Masking. We randomly pick a life-sequence and mask the background information (i.e., sex, origin, and birthday) with the `[UNK]` token.

Token Downsampling. We randomly pick a life-sequence and randomly remove tokens from the life-events. This procedure does not affect `[CLS]` and `[SEP]` tokens.

The augmentation procedures are independent of each other. Thus, some sequences might be altered by multiple procedures—the order of application: Sequence Downsampling, Temporal Noise, Background Masking, and Token Downsampling.

E.3 Interpretability

Attention Score When we use the Pooled Attention Decoder (see SI Sec. E.1, we can extract the attention weights associated with each concept token in a sequence.

Saliency score indicates the degree of change (and is directly connected to the partial derivatives). The contribution is calculated by back-propagating through the network, starting from the output score toward each token in the sequence. The higher the gradient associated with a token, the higher the contribution towards the predicted values since small changes in a particular concept token embedding would lead to a higher degree of change in the output. To achieve robust scores and minimize the noise associated with the gradient descent, we use `SmoothGrad` implementation [34] of saliency. x_i is an embedding of a token, $f(x_{1:n})$ is the output of the model, n is the number of noisy samples.

$$S(x_i) = \frac{1}{n} \sum_{n} \left\| \frac{\partial}{\partial x_i} f(x_{1:n}) x_i + e \right\|_2, \quad e \sim \mathcal{N}(0, \sigma^2)$$

Attention scores and Saliency Scores help us understand how big of a contribution each token towards the final prediction. The example is shown in a Fig. A.2.

Discovery of the concept directions via TCAV method. The following provides a step-by-step workflow to (1) find the direction of the concept (2) evaluate these directions:

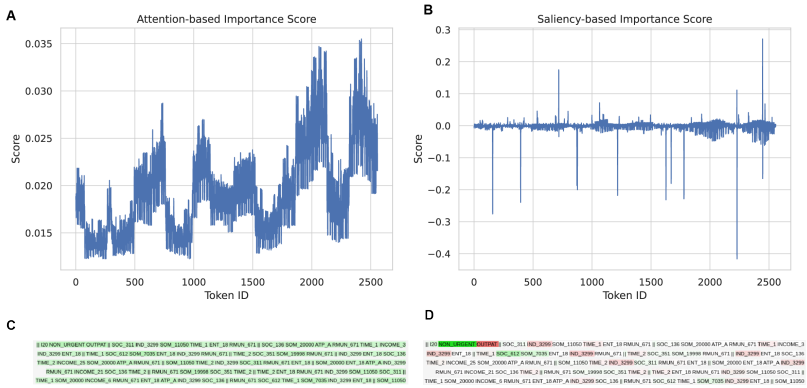

Figure A.2: Visualisation of Attention and Saliency Scores for a specific sequence (only small piece of a sequence is shown). This sequence belongs to a person who survived the four-year interval, the model assigned the probability of an early mortality of 0.23. (**A** - **B**) Attention and Saliency score per each token in the sequence. (**C**) Part of a life-sequence, each word is colored based on the assigned Attention score. (**D**) Part of a life-sequence, each word is colored based on the assigned Saliency score. Saliency scores provide more comprehensive importance score: red stands for negative contribution, and green stands for the positive contribution. `OUTPAT` (outpatient visit) lowers the probability of early mortality, as well as `IND_3299` (Manufacturing activities). However, `SOC_612` (sick leave) increases the probability of death. The comparison between the Attention and Saliency scores supports the claim that Saliency is more meaningful importance score [35].

1. Specify a concept, $\mathcal{C}$, e.g. life-sequence contains at least one `E16` diagnoses over year 2015,
2. Randomly sample 10 000 life-sequences, $\mathbf{s}$, from the *test* dataset: (1) for every life-sequence, $\mathbf{s}$, find a person-summary, $h(\mathbf{s}) = \mathbf{x}$ (where h is the encoder part of the model), and (2) calculate the gradient of output values, $\nabla f(\mathbf{x})$, with respect to the person-summary, $\mathbf{x}$, where f is a decoder part of the model that takes a person-summary, $\mathbf{x}$, as an input and outputs logits,
3. Randomly sample 3 000 life-sequences, $\mathbf{s}$, (from the *validation* dataset) that satisfy the specifications of a concept $\mathcal{C}$ and calculate the person-summary, $\mathbf{x}$ for each sample (we will refer to this set as $\mathcal{D}_{\mathcal{C}}$
4. Randomly sample 5 000 life-sequences (from the *validation* dataset) that do not satisfy the specifications of a concept, $\mathcal{C}$ and calculate the person-summary, x for each sample (we refer to this set as $\mathcal{D}_{\neg\mathcal{C}}$
5. Using stratified 5-fold cross-validation, find the optimal l2-regularisation parameter for the logistic regression on $\mathcal{D}_{\mathcal{C}} \cup \mathcal{D}_{\neg\mathcal{C}}$ datasets. The task for the logistic regression is to predict whether the life-sequence satisfies the concept $\mathcal{C}$.
6. Train 1 000 logistic regressions (with the optimal l2-regularisation parameter found in previous step) on the bootstrapped od $\mathcal{D}_{\mathcal{C}} \cup \mathcal{D}_{\neg\mathcal{C}}$ datasets,
7. Find the orthonormal vector to each separating hyperplane found by the logistic regression. These normals are concept activation directions, h_C.
8. For each gradient vector, $\nabla f(\mathbf{x})$, (step 1) find a dot product between every concept

activation direction, h_C (step 6). The average of these values is a sensitivity score of a model to a particular concept.

As a baseline, we specify a random concept (i.e., no specifications). Then, during steps 2 and 3, we randomly sample sequences. Then, we use the Mann-Whitney U test [101] to compare the distribution of the scores of baseline and distribution of the scores of a particular concept, $\mathcal{C}$.

E.4 Attention Mechanism

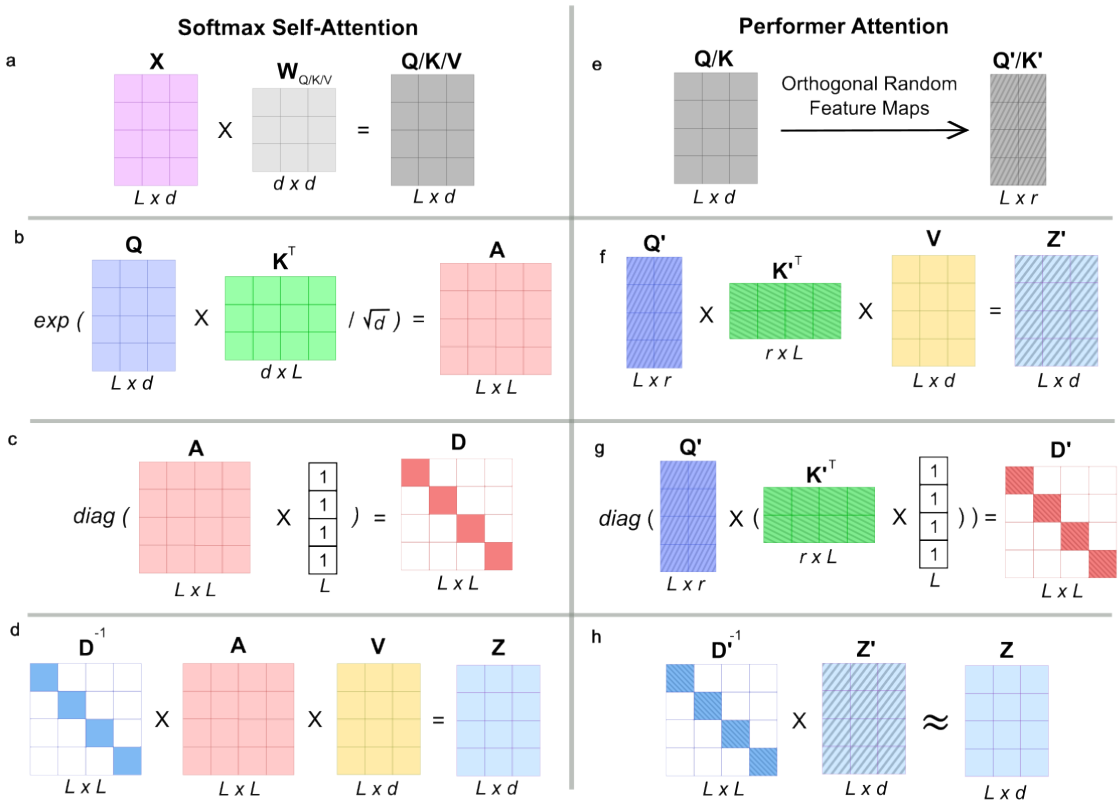

Figure A.3: Visualisation of the Attention Mechanism [52]. (**a-d**) Steps to compute the Softmax Self-Attention. (**a** and **e-h**) Steps to compute the FAVOR+ Attention (i.e., Performer-style Attention) [30] that approximates the Softmax Self-Attention output, the visualization omits the details for computing the Orthogonal Random Feature maps (for more details refer to [30]). You can find corresponding equations in Sec. 4.

E.5 Loss Functions

Based on the task, we use various loss functions to optimize the model.

Pre-training. Here we have two tasks – Masked Language Model (MLM) and Sequence Order Prediction (SOP). For the MLM, we use *cross-entropy loss*

$$\mathcal{L}_{CE}(\mathbf{y}, \mathbf{x}) = -\sum y_i \log(f(\mathbf{x})_i) \tag{A.1}$$

Here y is a one-hot encoded true target, and $f(x)$ is the output of the model. For the SOP task, we use weighted *cross-entropy loss with the label smoothing* [102]

$$\mathcal{L}_{CE-LS}(\mathbf{y}, \mathbf{x}) = (1-\alpha) \times -\sum w_i \, y_i \, \log(f(\mathbf{x})_i) + \frac{\alpha}{n} \times \sum w_i \, \log(f(\mathbf{x}_i)) \tag{A.2}$$

Here n is the number of classes, w_i is a weight assigned to class i, and α is scalar to control for a mixture of CE and Label Smoothing components (i.e., $w = [1.1, 10, 10]$, $n = 3$, and $\alpha = 0.1$). In Eq. A.1, $\forall\, i\, w_i = 1$.

Mortality Prediction and Emigration Prediction. We use *Asymmetric Cross-Entropy Loss* [59]. It accounts for the fact that the unlabeled set might contain positive samples. It also does not require any knowledge of a prior for the frequency of positive samples in the unlabeled set.

$$\mathcal{L}_{ACE}(\mathbf{x}) = -\frac{1}{p}\sum_{i=1}^{p}\log\left(g(f\left(\mathbf{x}_i\right))_1\right) - \frac{1}{n-p}\sum_{i=p+1}^{n}\log\left(g(f\left(\mathbf{x}_i\right) + \begin{bmatrix} c \\ 0 \end{bmatrix}^T)_0\right) \tag{A.3}$$

In Eq. A.3, $\mathbf{x}$ contains a batch of sequences, f is a model that outputs logits, $g(..)_1$ and $g(..)_0$ denote output (normalized scores) for positive and unlabeled classes (e.g. softmax function); while n is a total number of samples in a batch and p is the number of positive samples. $c \geq 0$ is a constant added to a logit of a unlabeled sample; we choose it by optimising AUL metric.

Predicting personality nuances. We use mixture of *Class Distance Weighted Cross-Entropy Loss* [90], *Focal Loss* [91] and *Label Smoothing* [102].

Class Distance Weighted Cross-Entropy Loss [90] handles imbalanced ordinal classification tasks. Instead of maximizing the likelihood of the true class, we minimize the likelihood of incorrect labels weighted by the absolute distance to the true class (Eq. A.4, where y is a true label of a sample (*not* one-hot encoded). CDW-CE has one hyperparameter – α, a distance penalty, which we set to 1.5.

$$\mathcal{L}_{CDW-CE}(y, \mathbf{x}) = -\sum_{i=0}^{N-1}\log(1 - f(\mathbf{x})_i) \times |i - y|^{\alpha} \tag{A.4}$$

Focal Loss [91] is another version of cross-entropy loss that focuses on the samples that are hard to predict (Eq. A.5), we set $\gamma = 5$. *Label Smoothing* (LS as in Eq. [92]) penalizes *overconfident* values (here $\mathbf{y}$ is a one-hot encoded target)

$$\mathcal{L}_F(\mathbf{y}, \mathbf{x}) = -\sum \log(\mathbf{y}_i - f(\mathbf{x})_i)^{\gamma}\log(f(\mathbf{x})_i) \tag{A.5}$$

The final loss function is presented in Eq. A.6. Without these modifications to the training procedure, the `life2vec` model converges to majority prediction. To train the model, we calculate $\mathcal{L}$ for each statement and then use the average of these losses to update the model.

$$\mathcal{L}(y, \mathbf{x}) = 0.3 \times \mathcal{L}_{CDW-CE}(y, \mathbf{x}) + \mathcal{L}_F(\texttt{onehot}(y), \mathbf{x}) + 0.1 \times \mathrm{LS}(\texttt{onehot}(y), \mathbf{x}) \tag{A.6}$$

E.6 Resampling based on the Instance Difficulty

We use the resampling strategy [89] as one of the methods to account for a large imbalance of classes (Personality nuances task). This particular method is described for the case when the sample has only one target. In our case, we have four (i.e., four items). Thus, we had to adapt the method. The difficulty of an ith sample after t steps is defined as [89]

$$D_{i,T} = c + \sum d_{i,t}$$

Given an ith sample, we calculate the difficulty $d^j_{i,t}$ according to Eq.4, 6, 7 in [89] (where t is a current epoch, and j is a jth item). In several cases, it might happen that the value of $d^j_{i,t}$ is extremely high - even if the predictions for this sample are good on the subsequent steps, the $D_{i,T}$ is still going to be large. Thus, we set a threshold for the value of $d_{i,t}$, which equals 100. The difficulty of the ith sample at the time step t is

$$d_{i,t} = \min(\max\{d^1_{i,t}, \ldots, d^j_{i,t}\}, 100)$$

After calculating all the difficulties at a step t, we apply robust scaling, where Qs are quantiles of difficulty scores at a step t

$$\text{RobustScaling}(d_{i,t}) = \frac{d_{i,t}}{Q_{0.75} - Q_{0.25}}$$

Lastly, we change the calculation of $D_{i,T}$ by introducing the Exponential Weighted Average ($\alpha = .5$ and $D_{i,0} = c$)

$$D_{i,T} = \alpha \cdot d_{i,t} + (1 - \alpha) \cdot D_{i,T-1}$$

These operations help to stabilize the sampling weights (for our multi-target case).

E.7 From Tabular Records to Life-Sequences

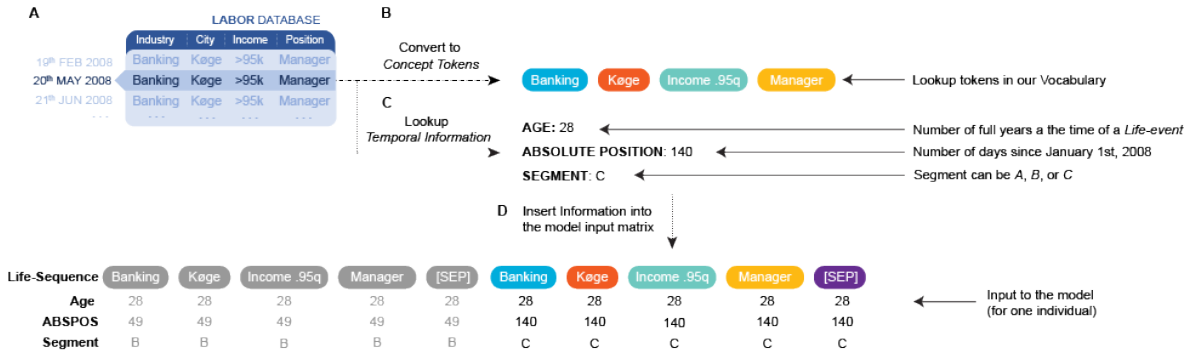

Figure A.4: The transformation of data from the Tabular format to the `life2vec` suitable format. (**A**) We start by looking up the next (chronologically) life-event in a person's history. (**B**) We convert relevant features to concept tokens of our vocabulary. (**C**) We lookup relevant positional information such as age, absolute temporal position, and segment of the life-event. (**D**) The information from the previous

F Visualisation of Embedding Spaces

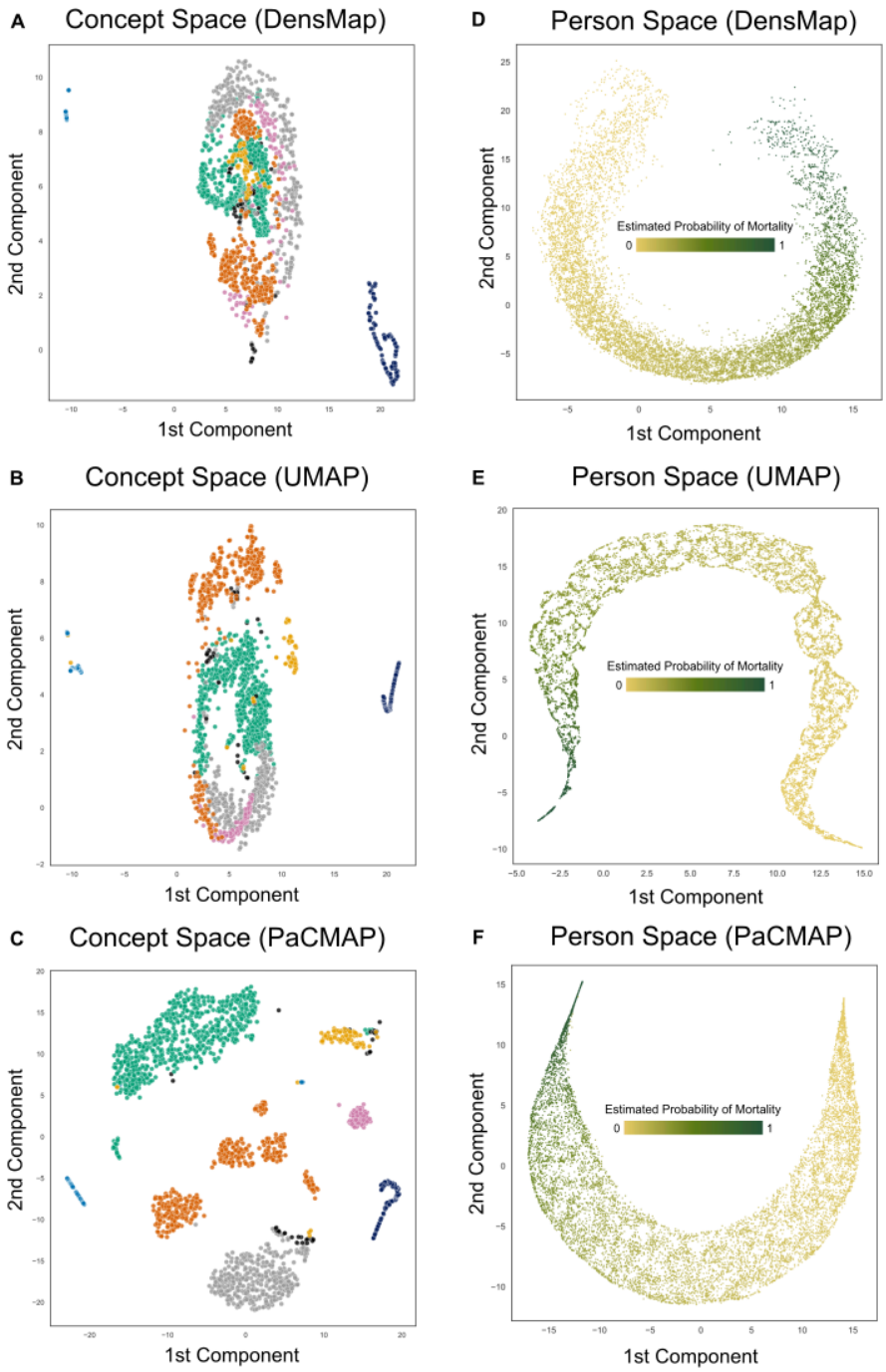

Figure A.5: Projections of Concept Embedding Space (**A**-**C**) and Person Embedding Space (**D**-**F**). We visualize each space with three different projection methods: DenseMap, UMAP, and PaCMAP. In (**A**-**C**), concept tokens are colored based on the category.

G Model Card: life2vec for Mortality Prediction

The model card refers to **two** models:

1. `life2vec` (*base*) model pre-trained on the unsupervised masked language modeling and sequence order tasks, and
2. `life2vec` (*mortality*) model finetuned on the Mortality Prediction Task.

G.1 Model Details

Person or organization developing model. `life2vec` (*base* and *mortality*) model is developed as part of the Nation-Scale Social Network Project. Specifically as part of the Ph.D. studies of Germans Savcisens at the Technical University of Denmark. The involved organizations are the Technical University of Denmark (Department of Applied Mathematics and Computer Science), Danmarks Statistik (Data Science Lab), and the University of Copenhagen (Copenhagen Center for Social Data Science).

Development period. September 2020 - April 2023

Model version. `v1` (*base*) and `v1` (*mortality*).

Model type. Transformer-based deep neural network.

G.1.1 Methods

Architecture details of the *base* model:

- **Architecture**: BERT-like encoder network
- **Optimization tasks**: Masked Language Modeling (MLM) and Sequence Order Classification (SOC).
- **MLM Decoder Network**: Two-layer neural network (pools representation of masked tokens). The final layer is tied to the Concept token embedding matrix (SI E.1).
- **SOC Decoder Network**: Two-layer neural network (pools `SEP` representation, SI E.1).
- **Attention mechanism**: Performer-style attention heads, with local softmax-attention heads (SI E.4).
- Other **architecture modifications** include ReZero & ScaleNorm normalization, Swish, Input-Output Embedding Tying, `time2vec`-based encoding of temporal information, sequence order prediction task, and cross-entropy loss with the label smoothing. These modifications ensure fast convergence and relatively small size of the model.
- **Optimization strategy**: AdamW Optimizer with the OneCycle LR Annealing. Trained for 30 epochs, where one epoch covers 30 000 randomly sampled (and augmented)

sequences from the training dataset.

Architecture details of the *mortality* model:

- **Architecture**: BERT-like encoder network (aka pre-trained *base* model)
- **Optimization task**: Binary mortality prediction task (early mortality within the next four years).
- **Classification decoder**: Two-layer network with the weighted averaging of concept representations (SI E.1).
- Other **architecture modifications**: Asymmetric Cross-Entropy (SI: E.5) for the Positive Unlabeled data sets and Sigsoftmax (as an alternative to Softmax).
- **Optimisation strategy**: RAdam optimizer with exponential LR annealing ($\gamma = 0.8$). Base LR for the decoder is 0.01, and LR for each consecutive encoder layer reduces by 5%. Token embeddings are frozen except for the `[CLS]`, `[SEP]`, `[UNK]` tokens. Temporal and Segment Embeddings are not frozen. For the decoder network, we set weight decay to 0.01. For the encoder layers, we set weight decay to 0.001.
- **Data Sampling**: We re-sample positive and negative samples to get approximately an equal fraction of both targets.

Fairness Constraints. None.

License: Not for public use or distribution.

Primary intended uses. The following information covers both **base** and **mortality** models:

- Use for scientific and research purposes only,
- Verify the validity of `NLP` inspired *Socio-economic* data representation,
- Verify the validity and performance of transformer-based architectures in the context of longitudinal socio-economic data
- Explore interactions between life-events and outcomes (i.e., mortality prediction) *on a global scale*.
- Use person-summaries as node-features in the Large-scale *Danish* population graphs
- Use person-summaries embeddings to study causal relationships between life-events.

Primary intended users. The following information covers both **base** and **mortality** models:

1. Denmark-based academics in Computational Social Science, Economics, Healthcare, Sociology, or Network Science.
2. Employees of Denmark Statistics (specifically Data Science Lab).

Out-of-scope use cases. The following information covers both **base** and **mortality** models:

1. **Not intended** as a tool to make judgments about specific individuals.
2. **Not intended** for a public release or deployment in governmental or private institutions.

G.2 Factors

This section describes the factors (e.g., groups, socio-economic attributes, sequence structure, etc.) that might lead to discrepancies in the model performance. **The section covers the *mortality* model.**

Potential **relevant factors** are

1. **Level of interaction with the healthcare system** – the fact that people use and consult healthcare providers with different frequencies (e.g., a person avoids interaction with the healthcare system or interacts only in severe cases),
2. **Socio-demographic attributes**: age, sex, and residency status (different groups, e.g., immigrants, ex-pats, natural-born citizens, and other residents, might have various access to public services and various sets of opportunities and limitations),
3. **Sequence Length** – longer sequences might contain more information (that model can use).
4. **Data drift and time** – we cannot guarantee the robust performance of the model beyond 2020 (e.g., COVID and human behavior).
5. **Cause of death**

Evaluation factors. Since **age** and **sex** is highly correlated with mortality outcomes, we want to evaluate the model's performance on unitary and intersectional splits of these groups (to probe the `life2vec`'s sensitivity to these features). Regarding the **residency status**, we are limited to the split based on the birthplace (i.e., in Denmark or outside). Thirdly, we want to evaluate how robust `life2vec` is regarding various life-sequences structures (aka length- and event-wise). Thus, we look at the **number of health-related events** in a sequence and the **length of the sequence**. Lastly, we do not have access to data beyond 2020. Thus, we cannot estimate the effect of the data drift on the `life2vec` model. However, we can evaluate how well the model predicts *distant* deaths.

G.3 Metric

Pre-training. We look at the perplexity score to evaluate and choose the most optimal `life2vec` (base) model (not presented in the model card).

Mortality Prediction. We frame the mortality prediction task as a positive-unlabeled problem. To optimize the `life2vec` (mortality) model, we use Area-Under-the-Lift (AUL), i.e., the early-stopping mechanism uses the AUL score. The primary performance evaluation metric for the `life2vec` model is Corrected Mathew's Correlation Coefficient (C-MCC) with a 95% Confidence Interval (we use bootstrapping). We use correction to account for the unlabeled samples in the test dataset. Along, we provide the corrected Balanced Accuracy and

Corrected F1-Score (refer to Tab. A8). All metric is reported at the .5 probability cutoff (not applicable to the AUL).

All the metrics presented in this model card are based on the test subset.

Table A8: Corrected Matthew's Correlation Coefficient (C-MCC) and Area under the Lift (AUL) on the Mortality Prediction Task (comparison of different baseline models). 95%-Confidence intervals for the MCC based on the stratified bootstrapping. In both cases, the higher value is preferred.

Model	MCC, 95%-CI	AUL	Accuracy, 95%-CI	F1-Score, 95%-CI	Model Size
L2V	**0.413 [0.410, 0.422]**	**0.845**	0.788 [0.782, 0.794]	0.443 [0.435, 0.451]	8.4m
RNN-GRU	0.369 [0.361, 0.378]	0.834	0.778 [0.771, 0.783]	0.395 [0.389, 0.402]	1.5m
FFNN	0.340 [0.332, 0.348]	0.822	0.768 [0.762, 0.774]	0.345 [0.339, 0.350]	8.4m
Logistic Reg	0.149 [0.142, 0.155]	0.735	0.639 [0.633, 0.645]	0.201 [0.198, 0.204]	2.0k
Life Tables	0.059 [0.051, 0.066]	0.650	0.555 [0.548, 0.562]	0.161 [0.158, 0.164]	3
Random	-0.005 [-0.011, 0.002]	0.497	0.496 [0.489, 0.503]	0.132 [0.128, 0.135]	-
Majority Class	0.0	0.497	0.5	-	-

Quantitative Analysis. We estimate the C-MCC score on the test data split (not the full one, but a random subsample of 20 000 people). See Fig. 6-9.

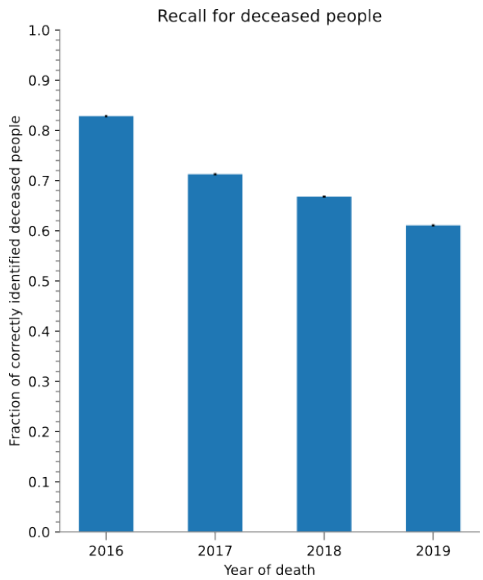

Figure A.6: The `life2vec`'s recall is based on the period between the day of prediction and day of death. The performance degrades as we get further away from the 31st of December 2015.

G.4 Data

Labor [11] and health data [13, 40] are provided by Danmarks Statistik (DST). These datasets include socio-economic, longitudinal information about the residents of Denmark. The use

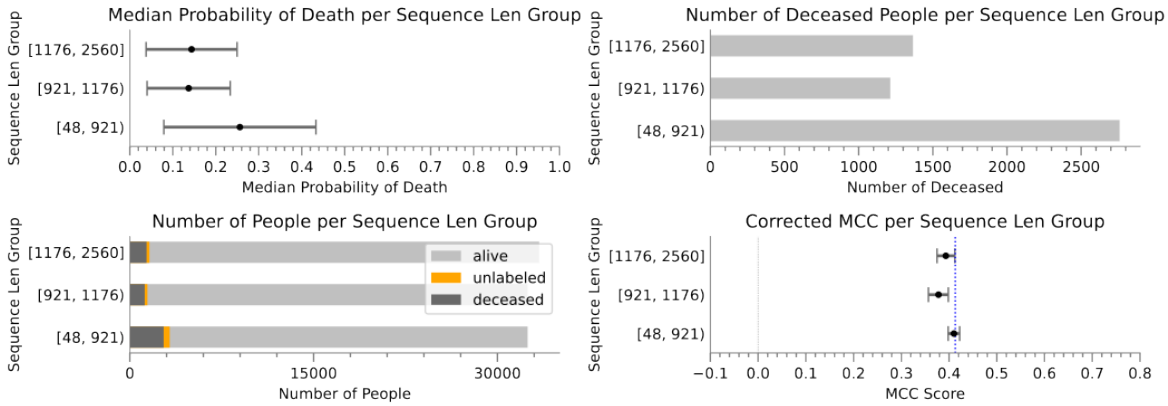

Figure A.7: Detailed evaluation of the `life2vec` model based on the **sequence length**. The length of the sequence does not seem to impact the performance of the model.

of the data [11, 13, 40] is regulated by (1) the EU Regulation on European Statistics, (2) the General Data Protection Regulation (GDPR), (3) the Danish Data Protection Act, (4) the Danish Public Administration Act, (5) the Danish Access to Public Administration Files Act, (6) the Danish Criminal Code, and (7) the Act on Statistics Denmark (DST). DST ensures that data of Danish residents and businesses is used **only for scientific purposes**.

Preprocessing. Refer to the Methods Section in the Original Paper.

Data split. We split data into training, validation, and test datasets (completely at random). **Training** subset is used to optimize the model. **Vaidation** subset is used to evaluate the model's performance at a specific epoch - we terminate the model's training if the performance metric on the validation data does not improve. **Test** subset estimates the final model performance.

Ethical Considerations. Refer to the Methods Section in the Original Paper.

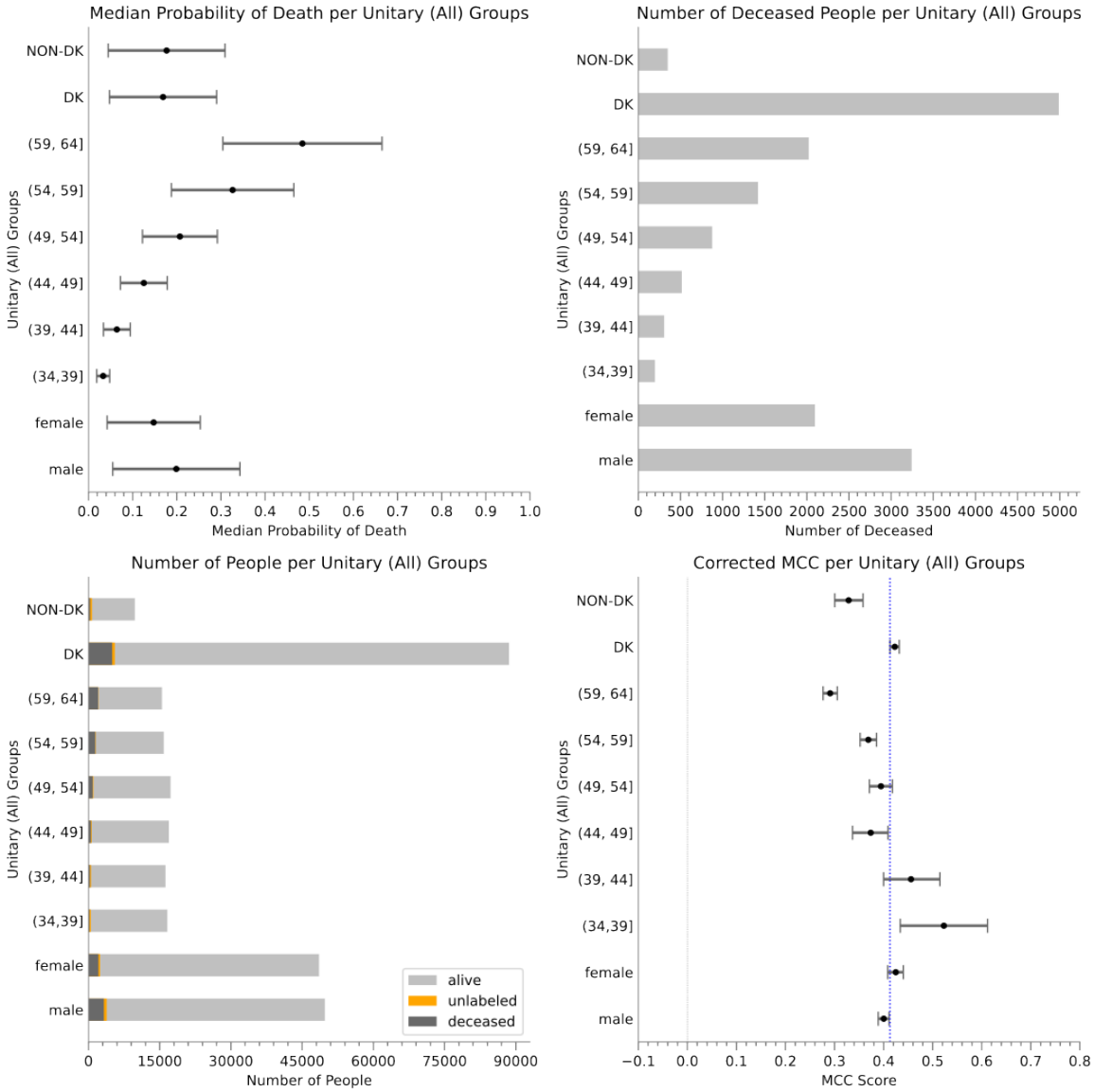

Figure A.8: Detailed evaluation of the `life2vec` model based on the **socio-demographic** attributes. The sequence length does not seem to impact the model's performance. **Age** – generally, older people have a higher probability of death. At the same time, the performance metric is worse for older people. **Sex** – the model's performance is similar regarding sex attributes. **Residency** – we can see a large difference between DK and NoN-DK groups, which might be connected to the imbalanced representation of groups.

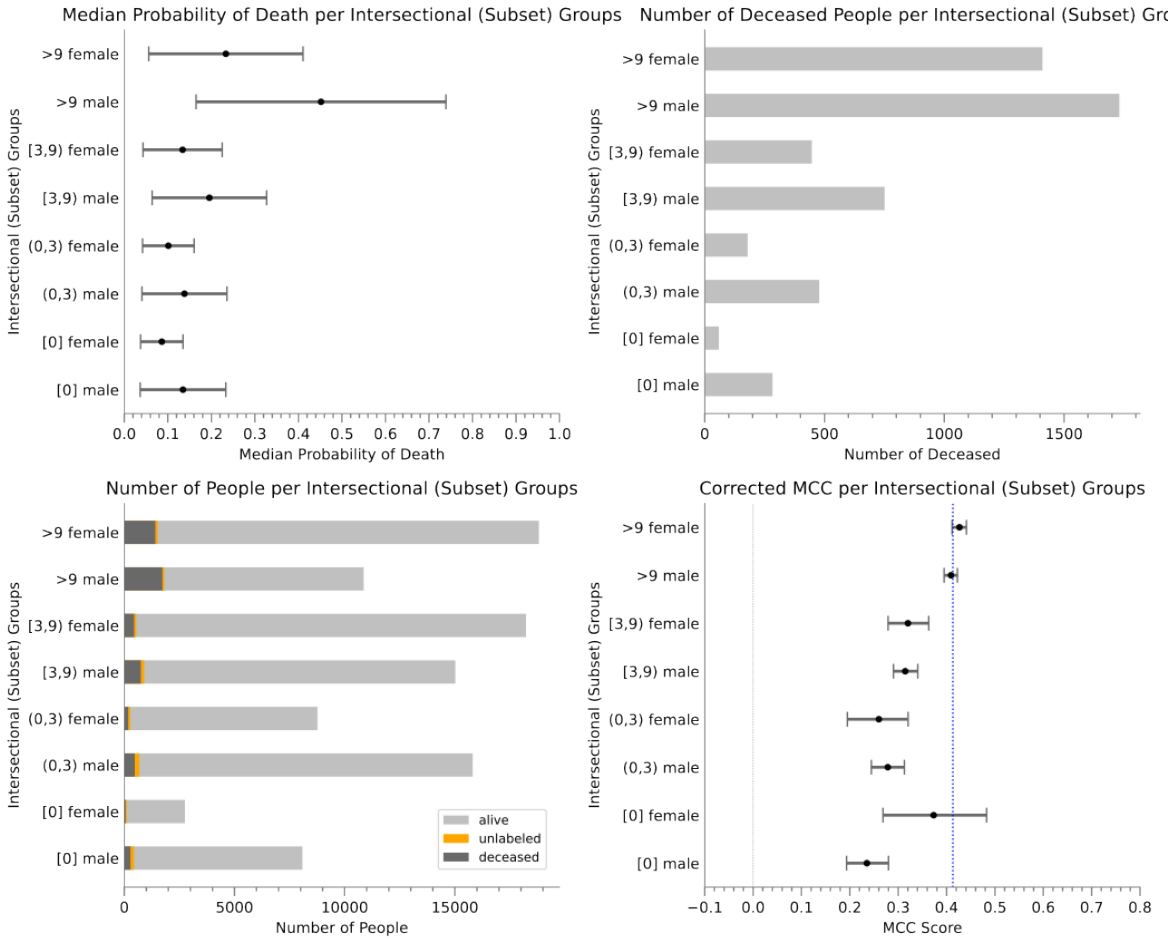

Figure A.9: Detailed evaluation of the `life2vec` model based on the intersection of **sex** and the **number of health events**. The results confirm that the level of interaction with the healthcare system *does have* an impact on the quality of predictions.