Analysis of DeepHit

Deep Learning Survival Analysis Algorithm

Andrew Cheng

Honours Mathematics Project

Mathematics and Statistics
McGill University
Canada

November 26th, 2019

Abstract

Survival analysis plays a critical role in extracting information from data in which the time to event is important. It is widely used in the financial, medical, engineering, and scientific fields. This paper presents the *DeepHit* algorithm developed by Lee et al., in-depth providing the background and intuition behind the model as well as presenting an unseen performance on a synthetic dataset. DeepHit is an incredibly flexible algorithm which assumes no parametric assumption allowing it to be applied in a multitude of different settings. DeepHit can also be applied in the *competing risks* environment. Besides the Fine-Gray model (Fine and Gray 1999), no survival analysis learning algorithm can be adequately applied to competing risks. DeepHit outperforms well-known state-of-the-art survival models such as Fine-Gray, cs-Cox, DMPG by a statistically significant amount. An algorithm like DeepHit allows one to effectively analyze survival data in which censoring occurs, an obstacle that has been long struggled with in the machine learning community.

1 Survival Data

In the survival analysis setting, one is concerned about the time until a specific event occurs, known as *time-to-event*. For example, time-to-event could be the time elapsed from the moment a patient is admitted to the hospital until his death (or recovery) from a particular disease. It could also be the time elapsed from the moment a brand-new phone is released to the public until the phone's defect. One is also concerned about the type of event that occurs at the time-to-event. For example, in the one-event setting, death is a common event of interest in the medical setting. When there are multiple events of interests, we call this the *competing risks setting.* In the time elapsed until phone defect example, an instance of competing risks could be the set of events - {water-damage, manufacturing defect, phone droppage}.

Definition 1. *We define $T = 0$ to be the origin of time, i.e., the time at which we start our study. We restrict our setting to discrete time intervals $T = \{0, T_1, \cdots, T_{max}\}$, where T_{max} is our predetermined maximum time horizon, i.e., the end of our study.*

Definition 2. *Let $\mathbf{K} = \{\emptyset, 1, \cdots, K\}$ be an ordered set of our competing events where $\emptyset$ denotes the occurrence of censoring. Let $t_i^{(j)}$ be the time that the i^{th} observation in the study experience event j. We define $s_i = min_{j\in\mathbf{K}}(t_i^{(j)})$ and $k = argmin_{j\in\mathbf{K}}(t_i^{(j)})$.*

Then a *Survival Triple* (x, s, k) constitutes a single survival data point, where x is a vector of covariates, s $\in T$ is the first time of event, and k $\in K$ is the specific event occurred at time s. Then it follows that $D = \{(x_i, s_i, k_i)\}_{i=1}^N$ is our survival data set for N observations. It is important to note that each observation can only experience one specific event.

Example 1. *For example, consider a study of N patients admitted to a hospital. Each patient is being admitted to a hospital at* $T = 0$ *(origin of time is relative to each patient). Let* $\mathbf{K} = \{\emptyset, \textit{heart disease, cancer}\}$ *be the cause of death. Then* $x^{(i)}$ *denotes the* i^{th} *patient's recorded covariates such as gender, age, etc.* $s^{(i)}$ *illustrates the time at which the patient dies and* $k^{(i)}$ *denotes the cause of death.*

Another important take away is that the occurrence of a competing event results in the censoring of all other competing events. For example, if patient i dies of a heart attack at time s_i, but would have experienced cancer at some other time u_i where $u_i > s_i$, we ignore the occurrence of cancer. Also note that all datapoints that survive past T_{max}, where T_{max} is the pre-defined maximum time horizon, experiences the censoring event at time $C = T_{max}$, this is known as *type I censoring* or *right-censoring*. These points will be further illustrated when the *cause-specific cumulative incidence function* is introduced.

The goal at hand is to develop an accurate and precise predictor model that captures the distribution of time-of-events based on covariates in the data set. In other words, given a dataset of size N, we attempt to formulate an algorithm such that using each observed survival triple, $\{(x_i^*, s_i^*, k_i^*)\}_{i=1}^N$, we learn the probabilistic estimator $\hat{P}$ which estimates the true probability $P(s = s_i^*, k = k_i^* |\ \mathrm{x{=}x}_i^*)$. Thus given a new observation x^{new} not within our dataset, we can accurately estimate the true probability $P(s = s_i^{new}, k = k_i^{new} |\ \mathrm{x{=}x}^{new})$, in other words, estimate the true probability of event k_i occurring at time s_i conditioning on x^{new}'s covariates.

1.1 The Problem of Censoring

Censoring pose many complications to the statistical learning community. In the medical setting example introduced in the previous section, right-censoring may occur when patients drop out of the study due to unexplained factors. This results in unknown *hitting times* of the competing events thus losing valuable information for our model to learn from. One way to deal with censoring is to throw away all censored data points. Clearly this method is wasteful, especially in data-sets in which censoring frequently occurs. Survival analysis methods are then used to 'recover' knowledge hidden by censoring. In section 6, I will introduce the concept of ranking, a means to deal with censored data.

One key assumption commonly used is that *censoring is non-informative*. This means that being censored at some time C only tells us that the true time the event occurs, s, must happen after C, i.e $s > C$, and nothing else. This implies that the censoring times is independent of the time of events - $s \perp\!\!\!\perp C$. This is opposed to *informative censoring*. An example of informative censoring is the following: suppose medical patients had a tendency to drop out of the study as they reached the chronic stages of their disease (consequently right-censored before their deaths), then clearly there exists some bias that must be accounted for.

2 Deriving the Survival Likelihood

Before diving into the main algorithm, the survival likelihood setting will be introduced. This will motivate the need for a nonparametric algorithm like DeepHit.

2.1 Survival Cumulative Distribution

Suppose we are restricted to the one-competing risk setting. That is, let $K = \{\emptyset, \text{event}\}$ and T be a non-negative random variable that represents the time elapsed until either event or censoring occurs. Let $T \sim F$, for some distribution function F. Let f be the probability mass function then the survival cumulative distribution function is defined as:

$$F(t) = Pr(T \leq t) = \sum_{u=0}^{t} f(t)$$

In words, this distribution function tells us the probability in which either censoring or the event has occurred to an individual before some time $0 \leq t \leq T_{max}$. The complement of the distribution function:

$$S(t) = 1 - F(t) = Pr(T > t)$$

is known as the survival function which depicts the probability that no event nor censoring has occurred by time t.

2.2 Survival Likelihood

Theorem 1. *Let $T_1, \cdots, T_n \overset{IID}{\sim} F$. By assuming non-informative censoring and known censoring times C, we can derive the survival likelihood function for a given dataset $\mathcal{D}$ of size N:*

$$\mathcal{L}(\mathcal{D}|\theta) = \prod_{i=1}^{N} f^{\delta_i}(t_i) \cdot S^{1-\delta_i}(t_i)$$

*where $\delta_i = 1$ if the i^{th} observation does **not** experience an event/censoring by time t_i and $\delta_i = 0$ if it does experience event/censoring by time t_i.*

Proof. We define $s_i = min(T_i, C_i)$ and $\delta_i = \mathbf{1}(T_i \leq C_i)$, where $\mathbf{1}$ is the indicator function. By our assumptions, we have that the binary bivariate random variables (s_i, δ_i) are IID for $i \in \{1, \cdots n\}$. We have that $(s_i, 0)$ only if $T_i > C_i$. Hence $P\{(s_i, 0)\} = P(T_i > C_i) = S(s_i)$. Also, we have $(s_i, 1)$ only if $s_i = T_i$ which implies that $P\{(s_i, 1)\} = f(u_i)$. We have then

$$\mathcal{L}(\mathcal{D}|\theta) = \prod_{i=1}^{N} P\{(s_i, \delta_i)\} = \prod_{i=1}^{N} f^{\delta_i}(t_i) \cdot S^{1-\delta_i}(t_i)$$

□

The survival likelihood function is the product of the likelihood of each datum where each datum is either uncensored or non-informatively censored. Passing the likelihood to the negative logarithm and replacing the Dirac-delta function δ_i with an indicator function we achieve:

$$-\sum_{i=1}^{N}[\mathbf{1}(k^{(i)} \neq \emptyset) \cdot log(f(t_i) + \mathbf{1}(k^{(i)} = \emptyset) \cdot log(S(t))]$$

which is one of the loss functions of DeepHit.

2.3 Cause-Specific Cumulative Incidence Function (CIF)

Now we introduce the *Cause-Specific Cumulative Incidence Function (CIF)* which is a generalization of the SCDF in the multi-competing events setting that also accounts for individual's covariates. In particular the CIF is a joint conditional probability distribution that depicts the probability a certain event k^* occurs before time t^* conditioning on the covariates x^*. This definition is fundamental in understanding competing risks in the survival analysis setting. Consider the CIF for the event of interest k^*, conditioning on the covariates $x = x^*$.

$$\begin{aligned} F_{k^*}(t^*|x^*) &= P(s \leq t^*, k = k^*|x = x^*) \qquad (1) \\ &= \sum_{s^*=0}^{t^*} P(s = s^*, k = k^*|x = x^*) \end{aligned}$$

where $F_{k^*}(t^*|x^*)$ is our CIF. Since the CIF distribution is not known in practice, we rely on the *estimated* CIF (ECIF), $\hat{F}_{k^*}(t^*|x^*)$. The CIF plays an important role in assessing the chance a particular competing event will occur conditioning on an observation's covariates. This function will help drive DeepHit to assess its performance in discriminating across cause-specific risks conditioning on a set of covariates. Note that the CIF accounts for the fact that the first competing event results in the censoring of all other competing events. Taking the log-likelihood of the CIF as I did with the SCDF and by taking the negative of that, will result in one of the loss functions of DeepHit, namely:

$$\mathcal{L}_1 = -\sum_{i=1}^{N}[\mathbf{1}(k^i \neq \emptyset) \cdot log(\hat{P}(s^{(i)}, k^{(i)}|x^i)) + \mathbf{1}(k^i = \emptyset) \cdot log(1 - \sum_{k=1}^{K} \hat{F}_k(s^i|x^i))]$$

where we denote $\hat{P}(s, k|x)$ as the probability that an individual will experience the event k at time s conditioning on a set of covariates x.

3 Multitask Learning Explained

Multi-task learning is the ability to learn different tasks in a single model. In *DeepHit* each *task* corresponds to a competing risk event. In other words if our data has K competing risks, the architecture of DeepHit will have K different tasks to learn.

The intuition behind multi-task learning for DeepHit is two-fold. First of all, many competing risks have shared knowledge and representations, in particular these risks may be correlated or even dependent on one another. For example, drugs that treat breast cancer will raise the risk of heart disease which means these two diseases are not independent (Koene et al. 2016) when breast cancer is being treated. By using multi-task learning, DeepHit exploits training signals from learning related tasks which tend to be neglected by other machine learning techniques. This allows for increased generalization of our algorithm, specifically *simultaneous joint learning* of the signals of our K competing events allows DeepHit algorithm to pick up both the subtle noises specific to each event and the underlying relationship (if any) between the events themselves. In addition, this enables the algorithm to learn the joint distribution rather than the marginal one.

Second of all, multi-task learning allows one model to perform different tasks which is practical for production systems. In other words, we can use one DeepHit model for a survival Dataset with a focus on K events, where K can be any reasonable number of competing events. This prevents the need to create a new model whenever we want to restrict or increase the number of competing events K.

4 Model Explained

We are interested in the joint distributions of K competing events and not the marginal distributions of each event. Thus a natural deep learning architecture to implement would be the *multitask neural networks* with a softmax output layer, which is exactly what the DeepHit algorithms is.

Architecture:

The structure of our deep learning network comprises of multitask learning with *hard parameter sharing.* Hard parameter sharing allows us to share the hidden layer of the network between all the cause-specific sub-networks tailored to each competing event. It has been shown that hard parameter sharing reduces risk of over-fitting.[1]

The DeepHit architecture has *K+1* sub-networks. One *shared sub-network*, L_S and K *Cause-Specific Sub-networks*, $L_{C,K}$, one for each competing event. The shared sub-network takes a vector of covariates x as input and outputs a common vector $f_s(x)$ which all cause-specific sub-networks take as input. The reason DeepHit feeds each cause-specific sub-network the same input when given an observation's covariates is to allow each cause-specific sub-network to learn the hidden underlying patterns in the covariates of x. This improves DeepHit's ability to differentiate the extent to which covariates affect which competing events and whether the presence of one covariate will diminish the effect of another covariate on a certain competing event.

DeepHit to feed each cause-specific sub-network the same input when given an observation's covariates. Each sub-network is connected to its own output layer which is part of the softmax output layer that enables the output of the joint probability of the K competing events. In general, a softmax output layer takes in a K-dimensional vector $\mathbf{Z}$ and outputs a K-dimensional vector Y of real-values between 0 and 1. Specifically, the output layer takes in the vector $\mathbf{Z} = \{z_1, ..., z_K\}$ where z_i corresponds to the output of the cause-specific sub-network i and outputs the joint probability distribution:

$$\mathbf{y} = [y_{1,1}, ...y_{1,T_{max}}, ..., y_{K,1}, ..., y_{K,T_{max}}]$$

where $y_{k,s} = \hat{P}(s, k|x)$.

In summary, the DeepHit learns the joint distribution of K competing events (rather than the marginal distributions of each event) by learning the shared representation of the events through its *cause-specific subnetworks*. In addition by using a soft-max layer, DeepHit maps the output onto a proper probability measure. Overall, the network produces an estimated joint probability distribution of the first hitting times of the competing events.

Loss Functions:
If more than one loss function is present in a deep learning algorithm, we say this is a multi-learning network. Thus DeepHit is a *multi-learning network* as DeepHit contains two loss functions $\mathcal{L}_1$ and $\mathcal{L}_2$. We define $\mathcal{L}_1$ as:

$$\mathcal{L}_1 = -\sum_{i=1}^{N}[\mathbf{1}(k^i \neq \emptyset) \cdot log(y^i_{k^i,s^i}) + \mathbf{1}(k^i = \emptyset) \cdot log(1 - \sum_{k=1}^{K} \hat{F}_k(s^i|x^i))]$$

where $\mathbf{1}(\cdot)$ is the indicator function and $\hat{F}_k(s^*|x^*) = \sum_{m=0}^{s^*} y^*_{k,m}$ is the *estimated cause-specific cumulative incidence function* introduced in the previous section. $\mathcal{L}_1$ is the negative log-likelihood of the joint distribution of the first hitting time and and corresponding event.

The first summation term accounts for instances where a non-censoring event occurs to the observation. This first summation term records both the time and which event has occurred to observation i.

The second term in the summation recovers the knowledge that if an observation is censored at time s then observation i has not experience an event up to s and some event $k \in K$ will occur after. In other words the first hitting time of each event $k \in K$ will occur after the censoring time s.

Together these terms drives representation learning in DeepHit, in particular to accurately estimate the true representation for the joint distribution of first hitting time and events conditioning on covariates.

5 Ranking

The second loss function is motivated by the concept of ranking. Ranking is a technique used to deal with censorship in survival analysis. How does one extract information from censored data?

Consider a medical study that investigates the probability of dying when suffering from a chronic disease. Suppose we conduct n number of checkups. Then we have T = $\{0, T_1, ..., T_n\}$, where T_i corresponds to the time of the i^{th} checkup. Prior to each checkup i, several patients are censored because they moved to another country rendering them unavailable for the rest of the clinical trial. These patients are said to be censored for this period of observation, with censoring time C_i. Also consider the case where the study concludes but there exists surviving patients. Obviously these patient's true survival times exceed this final observation period but yet these patient's will have censoring time C_n. Instead of ridding all these censored data points, we keep these data points and extract information when we can. The extent to which we can 'extract information' is dependent on how many *valid ordered ranking pairs* exist in our data. These ordered ranking pairs allow us to test how well our survival model performs, which will be further explained in section 6.

In the one-event setting, we say survival data points (i, j) are a *valid ordered ranking pair*, if one of the following two conditions hold: (1) if both i and j are uncensored data points with $s_i < s_j$ or (2) if the censored survival time of i is greater than the uncensored survival time of j, i.e $C_i > s_j$. Naturally, we can generate a corresponding directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes corresponding to survival data points and $\mathcal{E}$ is the set of directed edges. An edge $(u, v) \in \mathcal{E}$ exists if and only if (u, v) is a *valid ranking pair* meaning than u is *not* censored and the survival time of u is less than or equal to v, i.e. $s_u \leq s_v$. That is to say there are no outgoing edges originating from censored data points.

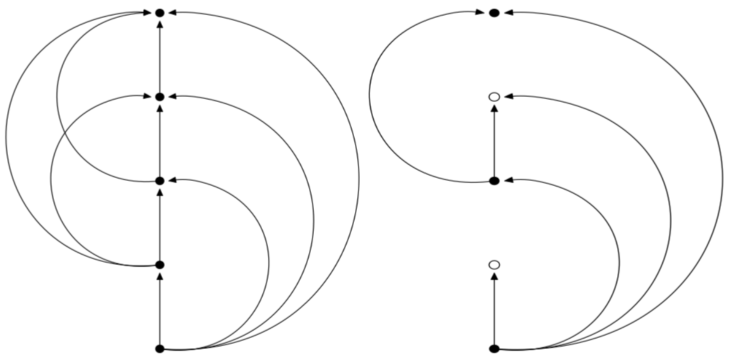

Figure 1[2]: Valid Order Ranking pairs formulated as a directed graph.(a) considers uncensored data and (b) considers censored data.

Ahuja and Schaar (cite) has shown that the concordance index is an ineffective metric when the number of competing events is greater than one. Instead, they propose the *joint concordance index*, a consistent estimator that accounts for the censoring bias. In short, the concordance index depicts the probability in which a fitted model accurately predicts the ranking of a subject among other subjects *while* correctly predicting the event type for the subject. However, these aforementioned metrics depend solely on the ordering of the predictions and thus useful in measuring the accuracy of Cox proportional-hazard models. This is because the Cox model assumes constant hazard ratios and therefore the ordering of proportional hazard models do not change over time. In the setting of DeepHit, a non-proportional hazard model, it is not trivial see how the concordance is applied in this setting. The metric used to evaluate DeepHit's performance is the time-dependent concordance index.

5.1 Second Loss Function

The second loss function drives the DeepHit to reward valid ranking pairs In other words,

$$A_{k,i,j} = \mathbf{1}(k^i = k, s^i < s^j)$$

is the indicator of pairs *(i,j)* such that $(i,j) \in E$. In other words, $A_{k,i,j} = 1$ when (i,j) is a valid ranking pair and 0 otherwise.

Now we have

$$\mathcal{L}_2 = \sum_{k=1}^{\mathcal{K}} \alpha_k \sum_i A_{k,i,j} \cdot \zeta(\hat{F}_k(s^i|x^i), \hat{F}_k(s^i|x^j))$$

α_k are constants used to trade off ranking losses of k^{th} competing event and $\zeta(x,y) = exp(\frac{-(x-y}{\sigma})$ is a loss function. minimizing $\mathcal{L}_2$ we drive DeepHit algorithm to maximize the number of valid ranking pairs.

6 Performance Metric

Changhee Lee, et al. uses the *time-dependent concordance index* ($C^{td} - index$), first introduced Boracchi, and Biganzoli (2005) to assess the performance of Deep-Hit. Before introducing C^{td}, recall the ordinary concordance index also known as the $C - index$.

6.1 C-index

Recall that given a survival dataset, a pair of survival points is a *valid ranking pairs* then either both data points are uncensored, or the uncensored time of one is smaller than the censored time of another. Also recall that we can generate a corresponding graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes corresponding to survival data points and $\mathcal{E}$ is the set of edges of $\mathcal{G}$, such that an edge $(u, v) \in \mathcal{E}$ exists if and only if (u, v) is a *valid ranking pair* and the survival time of observation u is less than or equal to v. Then using Raykar, et. al's definition, the *concordance index* is the fraction of all pairs of subjects whose predicted survival times are correctly ordered among all subjects that can actually be ordered. In other words the probability of concordance between the predicted and the observed survival:

$$CI(G, \hat{f}) = \frac{1}{|\mathcal{E}|} \sum_{(i,j)\in\mathcal{E}} \mathbf{1}_{\hat{f}x_i < \hat{f}x_j} \tag{2}$$

where $\hat{f}(x_i)$ is the predicted survival time for subject i by the model f. Note that this definition holds for the single event case.

6.2 Time-Dependent Concordance Index

DeepHit uses the time-dependent C-index, instead of the C-index. The time-dependent C-index estimates the probability that a pair of observations i and j are a valid ranking pair conditional that they are comparable,

$$C^{td} = P\Big(\hat{S}(T_i|x_i) < \hat{S}(T_i|x_j)|T_i \leq T_j, k_i \neq \emptyset\Big)$$

where $T_i = min\{T^*, C_i\}$ is the observation time and T^* is the true time of the i^{th} observation. Notice that the whole predicted survival function is utilized as outcome prediction (instead of the prediction of individual failure times), and the ability to discriminate among subjects having different outcome is summarized over time. Because C^{td} is independent of fixed time, it allows a proper assessment of settings in which survival times are effected covariates in a time-dependent manner. As per the C-index, C^{td} typically takes values between 0.5 and 1, where a score of 0.5 is as good as a random predictor model and a score 1 indicates perfect prediction accuracy. The performance metric of DeepHit uses the C^{td}-index which reflects the changes in risk over time. It provides an appropriate assessment for situation in which the influence of covariates on survival varies over time. One can see that that $C^{td} - index$ is equivalent to C-index when restricted to the one-event setting.

7 Experiment

7.1 Synthetic Data

I used Bellot and Schaar's' synthetic dataset[3] to demonstrate the ability of DeepHit to handle cubic and quadratic covariate effects. The synthetic model generates survival data with two competing events - $Z_i \in \{\emptyset, 1, 2\}$, and a possibility of censorship. I generated 5 datasets, each with 500 survival data points from the following:

$$X_i \sim \mathcal{U}(-2, 2)$$

$$T_i^1 \sim X_i^3 + \mathcal{N}(15, 1)$$
$$T_i^2 \sim 5X_i^2 + \mathcal{N}(5, 1)$$
$$T_i := \begin{cases} T_i^1, \text{w. prob } 0.8 \cdot I + 0.2 \cdot (1 - \mathbf{1}\{X_i \in (-1,1)\}) \\ T_i^2, \text{w. prob } 0.2 \cdot I + 0.8 \cdot (1 - \mathbf{1}\{X_i \in (-1,1)\}) \end{cases}$$

One can clearly see that quadratic and cubic behaviour is present among covariate effects on the generated dataset. The model generates data such that observations with covariates times in the interval (-1, 1) are more likely to experience cause 1 events. Similarly, observations with covariates times outside the interval (-1,1) are more likely to experience event 2. Furthermore, once the dataset is generated, 20% of the observed data is randomly selected to be censored. Censoring times C_i for observation i is selected by sampling from a uniform distribution in the interval between time 0 and the true hitting time, i.e, $C_i \leftarrow \mathcal{U}(0, T_i)$. This type of censoring is known as *random censoring*. The following figure depicts the synthetic generated times.

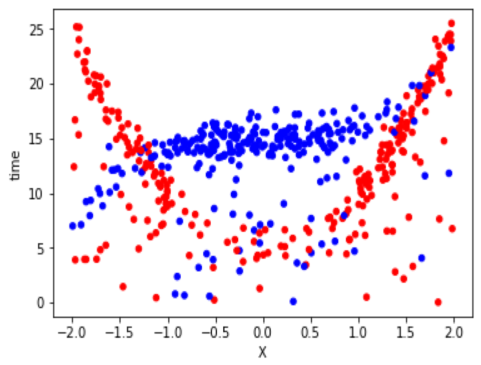

Figure 1: Blue and red colored objects correspond to observation from cause 1 and 2 respectively. The Y axis denotes survival time against the values of the covariate X.

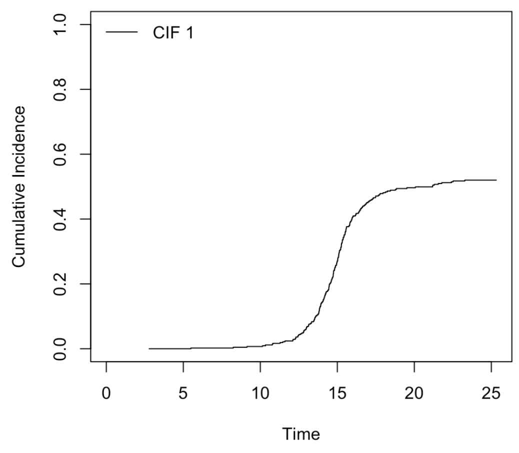

Figure 2: Cumulative Incident Function with respect to Event 1

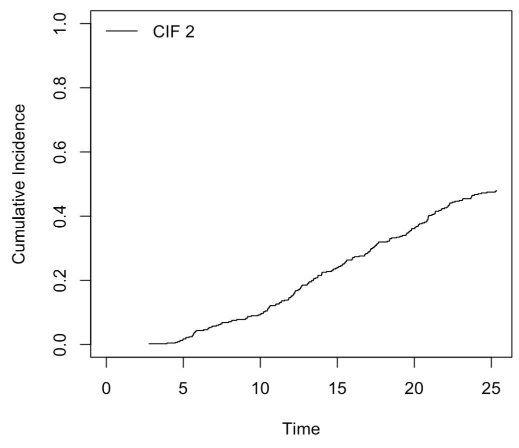

Figure 2: Cumulative Incident Function with respect to Event 2

8 Performance of DeepHit

The performance is based off applying 5-fold cross validation: randomly partitioning the data into a training set (80%) and a testing set (20%). In addition, 20% of the training set makes up the validation set. The hyper-parameter, α, was fixed at 0.85. A total of 5 synthetic data sets were generated and the following tables summarize the performance at the 90, 95 and 99 percent quantile of the hitting times when considering both events.

Trial (C_1)	90%	95%	99%
1	0.730±0.001	0.650 ±0.038	0.645 ±0.034
2	0.695 ±0.003	0.740 ±0.025	0.710 ±0.061
3	0.635 ±0.200	0.684 ±0.034	0.623 ±0.059
4	0.760 ±0.100	0.712 ±0.013	0.694 ±0.023
5	0.683 ±0.010	0.744 ±0.010	0.734 ±0.013

Table 1: Cause-specific C-index for event 1 evaluated at the 90, 95 and 99 percent quantile of hittine times

Trial (C_2)	90%	95%	99%
1	0.562±0.024	0.604 ±0.013	0.556 ±0.036
2	0.588 ±0.016	0.606 ±0.025	0.558 ±0.060
3	0.595 ±0.052	0.599 ±0.067	0.567 ±0.089
4	0.652 ±0.018	0.614 ±0.041	0.552 ±0.071
5	0.584 ±0.019	0.562 ±0.031	0.583 ±0.028

Table 2: Cause-specific C-index for event 2 evaluated at the 90, 95 and 99 percent quantile of hitting times

Algorithm	C_1	C_2
DeepHit	0.682 ± 0.047	0.563 ± 0.012

Table 3: Cause-specific C-index at the last observed time on the testing set of synthetic data. Uncertainty bands are standard deviations. The C-indices and their respective standard deviations averaged over the five data sets.

We observe that DeepHit has trouble dealing with predicting hitting times of event 2 compared to event 1. This is because the cumulative incidence of event 1 is flat for the majority of the time horizon and spikes up in in a narrow time frame. This is compared to the approximate linear rise of event 2's cumulative incidence across the majority of the time horizon. In other words, the hitting times of event 1 occur in a far more narrow time frame than that of event 2 enabling time 1 events to be much easier predicted.

9 DeepHit Compared to Other Survival Models

The Kaplan-Meier estimator is extensively used in clinical settings and fundamental statistical research in survival analysis. Although this estimator is able to learn very flexible survival curves, it is highly impersonal. The Kaplan-Meier estimator does not take into account patient's covariates. This means that the estimator acts on a population level rather than on an individual level. Furthermore, the Kaplan-Meier does not handle competing risks well. In particular, this estimator tends to overestimate absolute risks in the presence of competing risks.[4].

Another widely used survival model is the *Cox proportional hazard model*. It takes into account the patient's covariates, but restricts its flexibility through two main assumptions - (1) the hazard rate is constant and (2) the log of the hazard rate is a linear function of covariates. As a result this model is restricted to fewer settings in which it can learn the relationship between covariates and the survival parameters well, in particular it does not account for cases where the relationship between covariates and the parameters are time-dependent.

DeepHit counters these issues in two main ways: (1) it does not assume a form for the relationship between covariates and hitting times, in particular does not assume any kind of time-invariance, and (2): it learns the joint probability measure of survival times and events *directly*. In actuality, (2) implies (1) since by learning the joint probability distribution directly we do not need to assume any structure.

In Lee's paper, the authors reported experiments where DeepHit performed statistically significantly better than other state-of-the-art models in both multiple competing risk settings as well as setting in which there is only one event.

10 Future Work

DeepHit has outperformed many modern survival algorithms in various datasets. However, one could further test the robustness of DeepHit by generating an artificial dataset that exhibits a scenario in which an informative censoring assumption should be made. As mentioned earlier, DeepHit assumes an non-informative censoring assumption thus may be expected to experience performance issues under such situations. Another downfall of DeepHit is its inability to predict survival times exhibiting quadratic behaviour.

DeepHit assumes non-informative censoring in its data-sets. Performance on synthetic data-sets that illustrate informative censoring have not been reported, synthetic datasets that capture illustrative informative censoring can be used to further evaluate the robustness of DeepHit.

One concern is the way the authors generated their synthetic data-sets. They constructed two stochastic processes with parameters and hitting times described as follows:

$$x_1^{(i)}, x_2^{(i)}, x_3^{(i)} \sim \mathcal{N}(0, \mathbf{I})$$

$$T_1^{(i)} \sim exp\left((\gamma_3^T x_3^{(i)})^2 + \gamma_1^T x_1^{(i)}\right)$$

$$T_2^{(i)} \sim exp\left((\gamma_3^T x_3^{(i)})^2 + \gamma_2^T x_1^{(i)}\right)$$

where $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, x_3^{(i)})$ is the vector of clinical covariates for patient i and consists of three 4-dimensional variables: for $k = 1, 2$, the covariates x_k only have an effect on the hitting time for event k while x_3 has an effect on the hitting times of both events.

The problem lies in that the authors assume the hitting times are exponentially distributed with the mean parameter depending on the linear and non-linear (quadratic) function of covariates. In particular, the authors set $\gamma_1 = \gamma_2 = \gamma_3 = 10$. Setting the γ values to be 10 results in negative mean values which is an invalid parameter for the exponential distribution. The authors may have thrown out the data points that resulted in negative mean parameter. One should be careful in interpreting the synthetic results in the DeepHit paper.

One may look at the *Dynamic DeepHit algorithm* (Lee C., Yoon J., Van der Schaar M. (2018)) if dealing with longitudinal datasets consisting of various repeated measurements. This is compared to having only the latest available measurements. The algorithm is similar to DeepHit as it makes no parametric assumptions about the relationship between covariate effects and hitting times of the competing risks and has a familiar algorithmic architecture. But the algorithm is tailored towards incorporating longitudinal data-sets consisting of repeated measurements and has modifications to its loss functions and underlying structure accordingly so.

11 Citations:

1.Baxter, J. (1997). A Bayesian/information theoretic model of learning to learn via multiple task sampling. Machine Learning, 28, 7–39.
2.Raykar, V., Steck, H. On Ranking in Survival Analysis: Bounds on the Concordance Index.
3. Bellot, A., Van der Schaar, M. (2018). Tree-based Bayesian Mixture Model for Competing Risks
4. Lacny, S. et al. (2018). Kaplan-Meier survival analysis overestimates cumulative incidence of health-related events in competing risk settings: a meta-analysis.
5. Lee C., Yoon J., Van der Schaar M. (2018). Dynamic-DeepHit: A Deep Learning Approach for Dynamic Survival Analysis with Competing Risks based on Longitudinal Data