UNDER REVIEW

RESCIENCE C

Reproduction / Computational Biology

[Re] Improved survival analysis by learning shared genomic information from pan-cancer data

Christopher Huang[1] **and Wenhao Lu**[2]

[1]Heritage High School, California, United States – [2]Millburn High School, Millburn, NJ, United States

Edited by
(Editor)

Received
–

Published
–

DOI
–

Abstract We reproduce VAECox (Kim et al., Bioinformatics 2020), a transfer-learning survival model that pretrains a variational autoencoder on pan-cancer RNA-seq and fine-tunes its encoder inside a Cox proportional-hazards network. Using open-access TCGA data from the UCSC Xena hub (35 cohorts, 10,993 patients, 20,530 genes), we retrain the full pipeline and compare it with CoxLasso, CoxRidge, Cox-nnet and a Cox MLP over 10 random splits of the paper's 10 evaluation cancers. VAECox obtains the highest mean C-index (0.648 vs 0.638–0.646) and wins 5 of 10 cancers rather than the 7 reported; the per-cancer margins are one order of magnitude smaller than the seed-to-seed standard deviation, so the paper's ranking is only partially replicated. We extend the study with robustness, feature-budget, lightweight-model, subgroup-fairness and risk-stratification analyses.
A replication of S. Kim, K. Kim, J. Choe, I. Lee, J. Kang. Improved survival analysis by learning shared genomic information from pan-cancer data. Bioinformatics 36(Suppl. 1): i389–i398 (2020).

1 Introduction

Survival models trained on tumour gene-expression profiles suffer from a familiar imbalance: each cancer type contributes only a few hundred patients against roughly twenty thousand genes. Kim et al.[1] proposed VAECox to address this by learning *shared* structure across cancers first. A variational autoencoder (VAE)[2] is pretrained on pan-cancer RNA-seq from The Cancer Genome Atlas (TCGA)[3], and its encoder is then transferred into a Cox proportional-hazards network[4,5] and fine-tuned per cancer. The paper's central empirical claim is that VAECox outperforms CoxLasso[6], CoxRidge and Cox-nnet[5] on 7 of the 10 TCGA cancers evaluated, measured by Harrell's concordance index (C-index)[7].

The original code was released publicly (https://github.com/dmis-lab/VAECox) together with a 30-patients-per-cancer toy dataset, but the actual data came from the ICGC portal under controlled dbGaP access, which we could not obtain. Our replication therefore had two goals. First, verify that the released pipeline runs and behaves as documented (it needed several fixes to run on a current Python/PyTorch stack). Second, and more importantly, retrain the full method on *openly available* TCGA data from the UCSC Xena hub[8] and test the 7/10 claim with multiple random splits. Because VAECox is motivated by low-data regimes, we then extend the evaluation along axes the paper did not examine: robustness to corrupted inputs, accuracy under a restricted gene budget or a smaller VAE, whether such compression harms small cohorts disproportionately, performance gaps across clinical subgroups, and Kaplan–Meier risk stratification.

All code, result tables and figures are in our repository; the study proper is a single notebook that runs end-to-end on a Kaggle T4 GPU in roughly four hours.

Correspondence should be addressed to Wenhao Lu (wenhao@nxthorizon.org)
The authors have declared that no competing interests exist.
Code is available at https://github.com/Alscend-Research/survival-repro.

1.1 Scope: what we reproduce and what we do not

We reproduce the computational pipeline in full: pan-cancer VAE pretraining, per-cancer fine-tuning, the five-model comparison, and the temperature of evaluation choices around it. We do not reproduce the paper's biological interpretation of the learned latent space, and we make no claim about clinical utility. Within that scope we report three kinds of outcome and keep them separate throughout:

1. *Reproduced.* Qualitative claims that held: transferring a pan-cancer VAE encoder gives the best mean C-index across the ten cancers; the ordering of the deep baselines against the linear ones is broadly preserved; the pipeline trains end to end from released code on a single commodity GPU.
2. *Not reproduced numerically.* The 7-of-10 win count does not reproduce; we obtain 5 of 10, with per-cancer margins far below the seed-to-seed spread. We give our reading of why, and are explicit that the original single-split evaluation cannot settle it.
3. *New.* Findings beyond the original paper: the pretrained encoder is markedly more robust to noisy and missing expression values than a linear Cox model; accuracy degrades gracefully under a hundred-fold reduction in gene budget and a sixteen-fold reduction in VAE size; that compression does not disproportionately harm small cohorts; and that subgroup C-index gaps by age are as large for VAECox as for the baselines.

2 Methods

2.1 Original method

VAECox has two stages. (i) A VAE with a single hidden layer, $20{,}530 \to 4096 \to 128$ (mean and log-variance) and Tanh activations, is trained on the pooled expression matrix of all cancers with the usual reconstruction + KL objective. (ii) The trained encoder (μ head) is stacked under a Cox-nnet head (one hidden layer of 128 units) and the whole network is fine-tuned per cancer by minimising the negative Cox partial log-likelihood

$$\ell(\theta) = -\sum_{i:\,\delta_i=1}\Big[h_\theta(x_i) - \log\sum_{j\in R(t_i)} e^{h_\theta(x_j)}\Big],$$

where h_θ is the predicted log-hazard, δ_i the event indicator and $R(t_i)$ the risk set at time t_i. Baselines are a linear Cox layer with L1 (CoxLasso) or L2 (CoxRidge) penalty, Cox-nnet ($\sqrt{p}$ hidden units) and a two-layer Cox MLP (100 ReLU units, dropout), all trained with the same loss. The paper selects learning rate and weight decay per cancer by 5-fold cross-validation over 18 combinations and reports the test C-index.

2.2 Data

The paper used per-cohort FPKM matrices from ICGC. We instead used the TCGA `HiSeqV2_PANCAN` matrices and `clinicalMatrix` files from the UCSC Xena TCGA hub[8], as packaged in the open GenoTEX benchmark[9] (which lets the notebook run without any network access on Kaggle). These are $\log_2$-transformed, pan-cancer-normalised RNA-seq values. Thirty-five cohorts (10,993 patients) sharing 20,530 genes were loaded; all of them feed VAE pretraining, and the paper's ten evaluation cancers (BLCA, BRCA, HNSC, KIRC, LGG, LIHC, LUAD, LUSC, OV, STAD) are used for survival evaluation. Overall survival time and vital status were taken from the clinical matrix. Table 1 lists cohort sizes; the paper's gene set (20,502 genes, per-cohort normalisation) differs slightly from ours, which we note as a deviation.

Cohort	N	Events	Event rate	Cohort	N	Events	Event rate
BLCA	403	177	0.44	LIHC	365	130	0.36
BRCA	1080	151	0.14	LUAD	502	182	0.36
HNSC	517	220	0.43	LUSC	494	212	0.43
KIRC	531	175	0.33	OV	303	182	0.60
LGG	511	125	0.24	STAD	388	156	0.40

Table 1. The ten evaluation cohorts (UCSC Xena TCGA hub). Events are uncensored deaths.

2.3 Preprocessing, splits and evaluation

For each cancer and each of ten random seeds (0–9) we draw an 80/20 train/test split stratified by survival-time quintile, z-normalise every gene using training-set statistics only, and fit all five models on the training portion. Hyper-parameters (learning rate, weight decay) are chosen per cancer by cross-validation on the training set over a reduced grid of three (learning rate, weight decay) combinations, $(10^{-3}, 10^{-5})$, $(10^{-3}, 10^{-3})$ and $(10^{-4}, 10^{-5})$, rather than the paper's 18. Survival heads are trained for 100 epochs with Adam. The reported number is the test C-index averaged over the ten seeds (computed with `lifelines`[10]), and a "win" is assigned to the model with the highest mean on each cancer.

The VAE is trained once on all 35 cohorts for 500 epochs (Adam, learning rate 10^{-3}, weight decay 10^{-5}), matching the paper, except that we use mini-batches of 256 on the GPU rather than full-batch updates. VAECox is the paper's fine-tuned variant: encoder and Cox-nnet head are jointly trainable. Everything is implemented in PyTorch[11]; the model definitions and partial-likelihood loss are ported directly from the upstream repository. Runtime on a Tesla T4 is roughly one hour for VAE pretraining and 3.5–5 hours for the whole notebook; the notebook checkpoints the VAE and every result table so a run can be resumed across Kaggle sessions.

2.4 Extensions

All extensions use the same splits and fixed hyper-parameters (the per-model grid winner most often selected in the main run), so their absolute C-index values are close to but not identical with the main table.

- *Robustness.* Each trained CoxRidge and VAECox model is scored on test data with 0/10/25/50 % of gene values zeroed at random, or with additive Gaussian noise of $\sigma \in \{0, 0.5, 1, 2\}$ in z-score units (5 seeds).
- *Feature budget.* Only the top-k genes by training-set variance are kept, $k \in \{100, 500, 1000, 5000, 20530\}$; a k-input VAE is pretrained and both models are fitted (5 seeds, 3 cohorts). One configuration ($k = 1000$) is additionally forced onto CPU to measure accessibility.
- *Lightweight VAEs.* Hidden/latent sizes $4096/128$, $1024/64$, $512/32$, $256/16$, each pretrained for 100 epochs so configurations are compared at equal compute; we record the C-index change relative to the full model per cohort and correlate it with the cohort's number of events.
- *Subgroup fairness.* Test-set C-index within strata defined by age (≤ 50, 51–65, > 65), sex, stage (I–II vs III–IV) and histological subtype, for all five models, and the best-minus-worst gap per cancer×model×variable. Strata with fewer than 10 patients or 3 events are dropped, as are TCGA free-text placeholders ("Other, specify", "Mixed Histology (please specify)"), which are not clinical categories.

- *Risk stratification and interpretability.* Test patients are split at the median predicted log-hazard and compared with the log-rank test[12]; gene attribution for VAECox, whose first-layer weights are not gene-interpretable, uses permutation importance[13] (C-index drop when one gene's column is shuffled) on three cohorts.

2.5 Obstacles and deviations

Running the released code on Python 3.13 / PyTorch 2 without CUDA required seven small fixes (absolute-path resolution for the VAE launcher, deprecated NumPy/pandas calls, device handling); the upstream files are otherwise unmodified and the fixes are listed in our repository. The toy data shipped with the code (30 patients per cancer, as few as 3 events) can only verify that the pipeline executes: on it, VAECox won 4/10 cancers with a mean C-index of 0.572 versus 0.587 for the Cox MLP, but a single cohort (BRCA) swung between 0.00 and 1.00 across models on the same split, so we report those numbers only for provenance. The substantive deviations from the paper are: (1) open Xena data instead of controlled ICGC data, with a slightly different gene set; (2) a 3-point instead of 18-point hyper-parameter grid; (3) mini-batch instead of full-batch VAE training; and (4) results averaged over ten random splits rather than one.

3 Results

3.1 Reproduction of the main claim

Table 2 and Figure 1 give the test C-index of each model on each cancer. VAECox has the best mean across the ten cancers (0.648) and wins five of them (BLCA, LGG, LIHC, LUAD, OV); Cox-nnet wins three (BRCA, KIRC, STAD), CoxLasso one (HNSC) and the MLP one (LUSC). The paper reports seven wins for VAECox. Two features of the table matter more than the raw count. First, the mean C-index of all five models lies within 0.010 of each other (0.638–0.648), and the winning margin on individual cancers is typically 0.001–0.006. Second, the standard deviation across seeds is 0.02–0.08 on every cancer, an order of magnitude larger than those margins; in STAD the MLP and VAECox tie to four decimals. The paper's own Table 1 was based on a single split, and our reading is that the 7/10 count is largely within split noise: VAECox is competitive with, and on average marginally better than, the strongest baseline, but no model is consistently separable from the others on this benchmark. On LUSC and STAD every model is at or below 0.53, i.e. close to chance, in agreement with the low values the paper reports for these cancers.

Model	BLCA	BRCA	HNSC	KIRC	LGG	LIHC	LUAD	LUSC	OV	STAD	Mean
CoxLasso	0.628	0.700	**0.639**	0.732	0.815	0.703	0.608	0.501	0.590	0.515	0.643
CoxRidge	0.630	0.689	0.623	0.727	0.810	0.697	0.608	0.503	0.580	0.511	0.638
Cox-nnet	0.650	**0.714**	0.608	**0.753**	0.796	0.721	0.627	0.476	0.585	**0.530**	0.646
Cox MLP	0.651	0.679	0.605	0.731	0.815	0.688	0.643	**0.510**	0.573	0.516	0.641
VAECox	**0.655**	0.693	0.614	0.727	**0.822**	**0.727**	**0.644**	0.487	**0.595**	0.516	**0.648**
s.d. (VAECox)	.045	.057	.023	.029	.043	.036	.036	.024	.047	.069	

Table 2. Test C-index, mean over ten random 80/20 splits, real TCGA data. Bold marks the best model per cancer. The last row gives the seed-to-seed standard deviation of VAECox; the other models are similar (0.015–0.080). Wins: VAECox 5, Cox-nnet 3, CoxLasso 1, Cox MLP 1 (paper: VAECox 7/10).

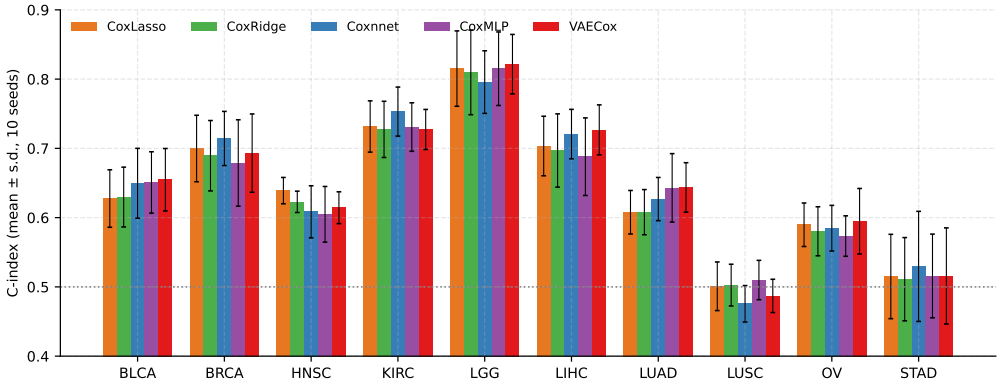

Figure 1. Per-cancer C-index with seed-to-seed standard deviation as error bars. The bars of the five models overlap on every cancer.

3.2 Robustness to missing and noisy input

Figure 2 averages over the ten cohorts. With half of all gene values zeroed, CoxRidge drops from 0.632 to 0.604 while VAECox is unchanged (0.631 to 0.630); with noise of $\sigma = 2$, CoxRidge falls to 0.589 and VAECox to 0.628. The effect is strongest on the cohorts where the encoder is useful to begin with (KIRC, LIHC, LGG: VAECox loses < 0.01 where CoxRidge loses 0.05–0.11) and absent on LUSC and STAD, where both models are near chance. This is the clearest advantage of the pretrained encoder in our study: the 128-dimensional bottleneck acts as a denoiser for the downstream Cox layer.

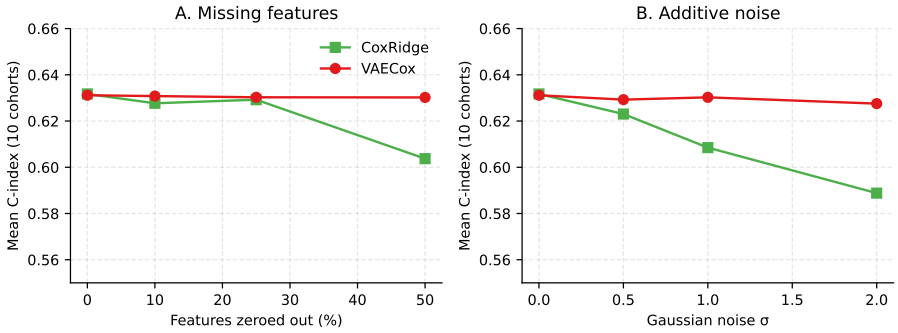

Figure 2. Mean test C-index over ten cohorts when (A) a fraction of gene values is zeroed or (B) Gaussian noise is added, for models trained on clean data.

3.3 Feature budget, lightweight models and CPU accessibility

Figure 3 shows that restricting the input to the 1,000 most variable genes costs VAECox 0.026 in mean C-index (0.638 to 0.612) but costs CoxRidge 0.074 (0.624 to 0.550), and that a 1.08M-parameter VAE on 1,000 genes pretrains on a laptop CPU in 11 s while reaching a C-index of 0.626 on the three test cohorts. Shrinking the full-gene VAE from $4096/128$ (170M parameters) to $256/16$ (10.5M) changes the mean C-index by less than 0.01 (Figure 4A). The change is not uniform across cohorts, but it is not systematically worse for small ones: averaged over the three reduced configurations, cohorts at or below the median event count move by -0.003 and those above it by $+0.005$ (Figure 4B). BRCA and OV lose the most (up to -0.06) and STAD gains the most (up to $+0.07$).

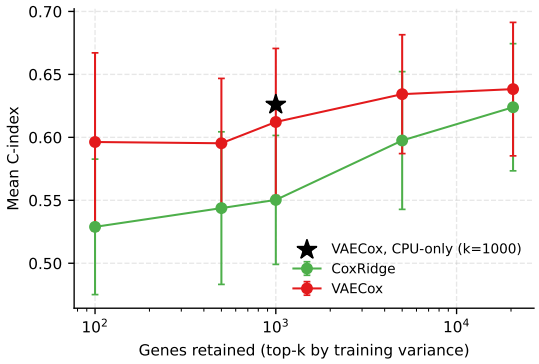

Figure 3. Mean C-index (three cohorts, five seeds) as a function of the number of genes retained; the star is a CPU-only run at $k = 1000$.

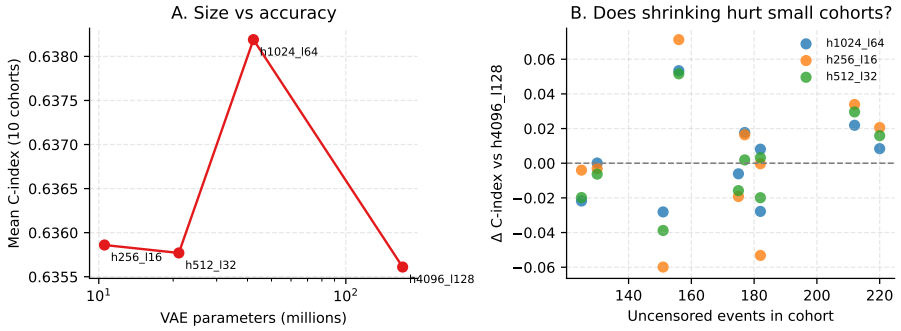

Figure 4. (A) Mean C-index versus VAE size, all configurations pretrained for 100 epochs. (B) Per-cohort change relative to the full model against the cohort's number of events; there is no downward trend for small cohorts.

3.4 Subgroup fairness

Across all cancer×model×variable combinations the mean best-minus-worst gap is 0.065 C-index. Age is the dominant axis (mean gap 0.099–0.100 depending on the model), followed by histological subtype (0.063–0.068), sex (0.053–0.055) and stage (0.026–0.036). The single largest gap is 0.242 (STAD, CoxRidge, patients over 65 versus 50 or younger). Before removing the two free-text histology placeholders in BRCA the largest gap was 0.475 and the mean subtype gap was 0.17; we mention this because a naive subgroup audit of TCGA would report a spurious disparity between two "unspecified" buckets. VAECox and the baselines show gaps of the same size, so pretraining neither creates nor removes subgroup disparities here.

3.5 Risk stratification and interpretability

Splitting the test set at the median VAECox log-hazard separates survival curves significantly (log-rank $p < 0.05$) in five of ten cohorts (BRCA $p = 0.006$, HNSC $p = 0.049$, KIRC $p = 3 \times 10^{-4}$, LGG $p = 0.026$, LIHC $p = 0.004$) and not in BLCA, LUAD, LUSC, OV or STAD—the same cohorts on which all models have C-index below about 0.65. Permutation importance on BLCA, BRCA and HNSC yields very flat profiles: shuffling any single gene reduces the VAECox C-index by at most 0.001, and the top-ranked genes (e.g. SCARB1 and CLEC3A in BRCA, CKS2 in HNSC) are not shared with the CoxRidge ranking. The encoder distributes its dependence over many genes, which is consistent with the robustness result and with earlier VAE analyses of TCGA transcriptomes[14], but it

means single-gene attributions for VAECox should not be over-interpreted.

4 Discussion

4.1 What the reproduction says about the method

VAECox reproduces in the way that matters most for a computational method: its pipeline runs unmodified from the released code, on data the original authors did not use, operated by people who were not involved in the original work, and the model it produces is the best of the five on average. What does not survive is the precision of the headline claim. A win count over ten cancers, computed from one split each, is a statistic with very little resolution: our seed-to-seed standard deviations (0.015–0.080) are ten to fifty times the margins that decide most of those wins. Re-running the same comparison with a different set of ten seeds would plausibly return a different count for any of the five models. We think the defensible version of the paper's claim is the one our data supports—that pan-cancer pretraining gives a small but consistent average advantage—and that the 7/10 framing overstates what the evaluation can resolve.

4.2 Where the reproducibility friction actually lives

None of our difficulties were with the model. They were with everything around it. The paper's data is behind dbGaP controlled access, so the exact input matrices cannot be obtained by a student, and the open Xena mirror we substituted has a slightly different gene set (20,530 versus 20,502) and a different normalisation. The released code had rotted against current NumPy, pandas and PyTorch, requiring seven small fixes before it would run at all. The toy dataset shipped alongside it is too small by an order of magnitude to test the claim it accompanies: on it, cohorts with three uncensored events return degenerate C-index values of exactly 0.00 or 1.00, and a reader who ran only the toy pipeline would conclude the method fails. Individually these are minor; collectively they describe the actual reproducibility surface of computational biology, which is dominated by data access and environment rather than by modelling.

4.3 Robustness, not accuracy, is the case for pretraining

The most useful thing we found is not in the original paper. Under corruption of the input the two model families separate far more cleanly than they do on clean data: at 50% of gene values zeroed, or at noise of $\sigma = 2$, the linear Cox model loses 0.03–0.04 C-index while VAECox loses essentially nothing. The same holds when the input is reduced to a thousand genes, where VAECox gives up 0.026 and CoxRidge gives up 0.074. If this holds on larger and more diverse sets, the practical guidance would be sharper than "pretraining buys accuracy": pretraining buys *tolerance*, and is worth most exactly where expression data is partial, noisy or cheap—which describes most settings outside a well-funded sequencing core. We regard this as a hypothesis our data supports rather than establishes.

5 Reproducibility statement

Everything needed to re-run this reproduction from a clean checkout is listed in Table 3. The study is a single notebook that runs end to end on Kaggle with the GenoTEX dataset attached; it checkpoints the pretrained VAE and every result table, so a run that exhausts a GPU session resumes where it stopped. The accompanying repository contains the C-index tables with per-cancer standard deviations, every extension result as a CSV, the

consolidated `manuscript_numbers.json` from which the tables in this paper are generated, a reproducibility card recording settings and deviations, and the toy-data track kept separately and labelled as pipeline verification only.

Item	Value
Original code	`github.com/dmis-lab/VAECox`
This reproduction	see *Code is available at*, front page
Data	TCGA via UCSC Xena hub, `HiSeqV2_PANCAN` (packaged in GenoTEX)
Cohorts	35 loaded (10,993 patients); 10 evaluated
Genes	20,530 shared
Hardware	Kaggle, Tesla T4, 1 GPU
VAE	$20{,}530 \to 4096 \to 128$, Tanh, Adam, lr 10^{-3}, wd 10^{-5}
VAE training	500 epochs, batch 256
Survival heads	100 epochs, Adam; per-cancer HP search, 3 combinations
Seeds	0–9 (all phases)
Splits	80/20, stratified by survival-time quintile
Models	CoxLasso, CoxRidge, Cox-nnet, Cox MLP, VAECox
Key packages	torch, lifelines, scikit-learn, pandas, numpy, matplotlib
Full pipeline	single notebook, ∼3.5–5 h on one T4
Table regeneration	single script, from result CSVs

Table 3. Reproducibility summary.

6 Limitations

1. Open Xena data rather than the paper's controlled-access ICGC matrices, with a slightly different gene set and normalisation. Absolute C-index values should not be compared to the paper's as if measured on the same thing.
2. A 3-point hyper-parameter grid instead of the paper's 18-combination 5-fold search, for compute reasons. All five models are affected equally, but every model may be sub-optimally tuned.
3. Extensions use fixed hyper-parameters rather than the per-cancer search, so extension C-index values are close to but not identical with the main table and should not be cross-read against it.
4. The robustness sweep uses five seeds and the lightweight sweep pretrains at 100 epochs rather than 500, so configurations are compared at equal compute rather than at convergence.
5. Ten cancers, and subgroup strata below 10 patients or 3 events dropped as unestimable. Several subgroup gaps rest on a few dozen patients.
6. Permutation importance was computed on three cohorts and two models only, and its profiles are too flat to support single-gene claims.
7. No experimental or clinical validation of any kind. Kaplan–Meier separation on held-out patients is the only outcome-level check.

7 Conclusion

VAECox's computational core reproduces. Using only open-access TCGA data, a reduced hyper-parameter search and a single GPU, we retrained the full pipeline and recovered the paper's qualitative picture: the transferred pan-cancer encoder gives the best mean

C-index across the ten evaluation cancers (0.648 against 0.638–0.646 for the baselines), and the released code required no modification to the model itself.

The specific claim that VAECox outperforms the baselines on 7 of 10 cancers did not reproduce: we measure 5 of 10. We regard the difference as largely an artefact of evaluation resolution rather than of implementation, since the per-cancer margins that decide the count are one to two orders of magnitude smaller than the variation across random splits—variation a single-split evaluation cannot reveal. That the original design could not have detected this is itself worth recording.

Beyond the paper, we find that the pretrained encoder is what makes the model robust rather than what makes it accurate: it holds its C-index under 50% missing values and $\sigma = 2$ noise where a linear Cox model loses 0.03–0.04, and it gives up a third as much as CoxRidge when the input is cut to a thousand genes. A 10.5M-parameter VAE performs within 0.01 of the 170M one, a 1.08M-parameter version trains on a laptop CPU in eleven seconds, and neither compression falls disproportionately on cohorts with few events. Age-related subgroup gaps reach 0.24 C-index and are no smaller for VAECox than for the linear models, which is an argument for reporting them routinely rather than an indictment of this method in particular.

For a method aimed squarely at the small-sample regime, the reassuring finding is that the thing it is supposed to buy—graceful behaviour when data is scarce, partial or noisy—is exactly what we measured. The concerning finding is that the claim it was published on cannot be checked by anyone without controlled data access, and would not have been stable if it could.

Acknowledgements

We thank the authors of VAECox for releasing their code and pretrained pipeline publicly, and the UCSC Xena team and the GenoTEX authors for maintaining the open TCGA mirrors without which this reproduction would not have been possible.

References

1. S. Kim, K. Kim, J. Choe, I. Lee, and J. Kang. “Improved survival analysis by learning shared genomic information from pan-cancer data.” In: **Bioinformatics** 36.Supplement_1 (2020), pp. i389–i398. DOI: 10.1093/bioinformatics/btaa462.
2. D. P. Kingma and M. Welling. “Auto-Encoding Variational Bayes.” In: **International Conference on Learning Representations (ICLR)**. 2014. URL: https://arxiv.org/abs/1312.6114.
3. The Cancer Genome Atlas Research Network, J. N. Weinstein, E. A. Collisson, G. B. Mills, K. R. M. Shaw, B. A. Ozenberger, K. Ellrott, I. Shmulevich, C. Sander, and J. M. Stuart. “The Cancer Genome Atlas Pan-Cancer analysis project.” In: **Nature Genetics** 45 (2013), pp. 1113–1120. DOI: 10.1038/ng.2764.
4. D. R. Cox. “Regression models and life-tables.” In: **Journal of the Royal Statistical Society: Series B** 34.2 (1972), pp. 187–202. DOI: 10.1111/j.2517-6161.1972.tb00899.x.
5. T. Ching, X. Zhu, and L. X. Garmire. “Cox-nnet: An artificial neural network method for prognosis prediction of high-throughput omics data.” In: **PLOS Computational Biology** 14.4 (2018), e1006076. DOI: 10.1371/journal.pcbi.1006076.
6. R. Tibshirani. “The lasso method for variable selection in the Cox model.” In: **Statistics in Medicine** 16.4 (1997), pp. 385–395. DOI: 10.1002/(SICI)1097-0258(19970228)16:4<385::AID-SIM380>3.0.CO;2-3.
7. F. E. Harrell, R. M. Califf, D. B. Pryor, K. L. Lee, and R. A. Rosati. “Evaluating the yield of medical tests.” In: **JAMA** 247.18 (1982), pp. 2543–2546. DOI: 10.1001/jama.1982.03320430047030.
8. M. J. Goldman et al. “Visualizing and interpreting cancer genomics data via the Xena platform.” In: **Nature Biotechnology** 38 (2020), pp. 675–678. DOI: 10.1038/s41587-020-0546-8.
9. H. Liu, S. Li, and H. Wang. **GenoTEX: A benchmark for automated gene expression data analysis in LLM-driven agent settings**. arXiv:2406.15341. 2024. URL: https://arxiv.org/abs/2406.15341.

10. C. Davidson-Pilon. “lifelines: survival analysis in Python.” In: **Journal of Open Source Software** 4.40 (2019), p. 1317. DOI: 10.21105/joss.01317.
11. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. “PyTorch: An imperative style, high-performance deep learning library.” In: **Advances in Neural Information Processing Systems 32**. 2019, pp. 8024–8035.
12. N. Mantel. “Evaluation of survival data and two new rank order statistics arising in its consideration.” In: **Cancer Chemotherapy Reports** 50.3 (1966), pp. 163–170.
13. L. Breiman. “Random forests.” In: **Machine Learning** 45 (2001), pp. 5–32. DOI: 10.1023/A:1010933404324.
14. G. P. Way and C. S. Greene. “Extracting a biologically relevant latent space from cancer transcriptomes with variational autoencoders.” In: **Pacific Symposium on Biocomputing**. Vol. 23. 2018, pp. 80–91. DOI: 10.1142/9789813235533_0008.