# Supplementary Materials

## 1 Data preprocessing

The datasets that were used in our experiments were obtained from:

- `https://huggingface.co/datasets/inria-soda/tabular-benchmark`
- `https://huggingface.co/datasets/marmal88/skin_cancer`
- `https://huggingface.co/datasets/stochastic/random\_streetview_images_pano_v0.0.2`
- `https://huggingface.co/datasets/Censius-AI/ECommerce-Women-Clothing-Reviews`
- `https://huggingface.co/datasets/james-burton/kick_starter_funding`

using the Python library `datasets`. Categorical variables were numerically encoded using `OrdinalEncoder` (`sklearn`) and the selected numerical variables were processed by log-transformation, or min-max scaling using `MinMaxScaler` (`sklearn`) (cf. Supplementary Table 1). No additional data processing was performed.

## 2 Implementation details

### 2.1 Encoders

To ensure a fair comparison across the methods, in all our experiments, we employed identical encoders. The architecture of each encoder was tailored to a specific variable type. Unless stated otherwise, the encoders were implemented as follows:

- **Categorical encoder** was implemented as a single embedding layer with the embedding dimension controlled by hyperparameter `encoder_dim`.
- **Numerical encoder** was implemented as a ReLU-activated MLP, with the following number of neurons {1, `hidden_dim`, `encoder_dim`}.
- **Image encoder** was implemented as a pre-trained ResNet50 [3], followed by linear projection to reduce output dimension to `encoder_dim`;
- **Text encoder** was implemented as a pre-trained BERT transformer [2], followed by linear projection to reduce output dimension to `encoder_dim`. The BERT tokenizer was used to tokenize the inputs before passing them to the encoder.

### 2.2 Models

The intermediate representation vectors, once produced by the respective encoders, were then processed through method-specific downstream computations, which were implemented based on the information from the respective papers, and, if applicable, using the provided code.

**Scarf** Representation vectors were concatenated and subsequently passed to a neural subnetwork $g$ (in the original paper referred to as the *pretraining head*). This subnetwork was implemented as a ReLU-activated MLP, with the number of neurons set to {`encoder_dim` $\times$ M, `encoder_dim`, `latent_dim`}; where is M is number of variables and `latent_dim` is hyper parameter denoting number of latent space dimensions. The contrastive loss was computed from the similarities between the resulting embeddings and the embeddings of the corrupted analogs. Note that when creating the corrupted views, unlike in the original implementation, we sampled from the batch marginals instead of dataset marginals. This is a minor change that allows substantially faster runtime when applied to image/text-tabular datasets.

**SubTab** Representation vectors were grouped into $k = 3$ overlapping blocks, with the 75% overlap. These blocks were subsequently concatenated and passed to a neural subnetwork $E$, implemented as a ReLU-activated MLP with the number of neurons set to {`encoder_dim` × M, `encoder_dim`, `latent_dim`} to produce block embeddings. The contrastive loss was computed from the similarities between the resulting embeddings; along the distance loss between the embeddings.

**CLIP** No further steps were applied. The intermediate representation vectors produced by encoders were used as the final embeddings, from which the pairwise contrastive loss was computed.

**GMC** Joint representation vectors were produced by passing inputs jointly through designated subnetwork $f^{(1:M)}$, which consisted of variable specific encoders analogous to those described above (without sharing weights), followed by linear projection to reduce dimensions to `encoder_dim`. Variable-specific and joint representation vectors were passed through *projection head* $g$, implemented as SiLU-activated (swished) MLP, with the number of neurons set to {`encoder_dim`, `encoder_dim`, `latent_dim`}, to produce their respective embeddings.

**MCN** The intermediate representation vectors produced by the encoders were averaged to, so-called, *fused multimodal features*, which were then subjected to online k-means clustering [1]. Parameter $k$ of the online k-means clustering was set to `batch_size`/20, but not less than 2.

**ICE-T** The neural subnetwork $g$ was implemented as a ReLU-activated MLP with the number of neurons set to {`phi_dim`, `phi_dim`, `latent_dim`}.

## 2.3 Hyperparameters

The hypermaramenters used in our experiments include: `hidden_dim`, `encoder_dim`, `phi_dim`, `latent_dim`, `learning_rate` and `batch_size`. The range of values tried per hyperparameter across the datasets in our experiments are available in the code supplement (`/src/config.yml`). The performance resulting from each hyperparameters configuration across the datasets can be found in code supplement (`/results/dataset_name.csv`).

## 2.4 Hardware & Software

All our experiments were performed using multiple `NVIDIA RTX A6000` GPU cards, `CUDA v12.2`, on `Ubuntu v20.04.6 LTS`. All the code was written in Python programming language and was run using `Python v3.8.10`, `torch v2.1.0`, `numpy v1.24.4`, `sklearn v1.3.2` and `pandas v2.0.3`. For more details see the `requirements.txt` file in the code supplement.

## 2.5 ICE-T

```python
class ICETEmbedder(nn.Module):
    def __init__(self, mappings: dict, phi_dims: list, hidden_dim: int, latent_dim: int, tau_init: float = 0.07):
        super(ICETEmbedder, self).__init__()
        # Base mappings
        self.mappings = nn.ModuleDict(mappings)
        # Phi function
        self.phi = MLP(hidden_dim, latent_dim, phi_dims)
        # Temperature
        self.tau = nn.Parameter(torch.log(torch.tensor(tau_init)))

    def sim_func(self, x, y):
        a = F.normalize(x, dim = 1)
        b = F.normalize(y, dim = 1)
        sim = torch.matmul(a, b.T)
        sim = sim / torch.exp(self.tau)
        return sim
```

```python

    def loss_func(self, sims):
        labs = torch.arange(sims.shape[0], device=sims.device)
        loss = F.cross_entropy(sims, labs, reduction='mean')
        return loss

    def calc_loss(self, H, h):
        # Sum latent representations
        H_sum = h * len(H)
        # Denominator
        k = len(H) - 1
        # Calc loss
        loss = 0.
        for hv in H.values():
            # Calc anchor
            mu = (H_sum - hv) / k
            # Project by phi
            zv, z_mu = self.phi(hv), self.phi(mu)
            # Calc similarities
            sims = self.sim_func(zv, z_mu)
            # Calc loss
            loss += self.loss_func(sims)
        return loss

    def transfer(self, query_var: str, candidates: torch.Tensor, evidence: dict):
        # Get latent representations of evidence
        H = [self.mappings[v](x) for v, x in evidence.items()]
        # Joint representation of evidence
        mu = sum(H) / len(H)
        # Get latent representation of query
        h = self.mappings[query_var](candidates)
        # Make projections
        z, z_mu = self.phi(h), self.phi(mu)
        # Calc similarity
        sim = self.sim_func(z, z_mu)
        # Select candidate value
        estimate = candidates[torch.argmax(sim, dim = 0)]
        return estimate

    def forward(self, X: dict, calc_loss = True):
        # Get interemdiate representations
        H = {v:mapping(X[v]) for v, mapping in self.mappings.items()}
        # Get joint intermediate representation
        h = sum([v for v in H.values()])/len(H)
        # Get joint representation
        z = self.phi(h)
        # Calculate loss
        if calc_loss:
            loss = self.calc_loss(H, h)
        else:
            loss = None
        return z, loss
```
Listing 1: Python implementation of ICE-T

# 3 Supplementary tables

| Dataset | Processing step | Variable |
|---|---|---|
| airlines | min-max scaling | 'CRSDepTime' |
| | | 'CRSArrTime' |
| | | 'Distance' |
| | | 'DepDelay' |
| allstate | min-max scaling | 'cont1' – 'cont14' |
| | | 'loss' |
| brazilian_houses | log-transform | 'hoa_(BRL)' |
| | | 'rent_amount_(BRL)' |
| | | 'property_tax_(BRL)' |
| | | 'fire_insurance_(BRL)' |
| | | 'total_(BRL)' |
| covertype | min-max scaling | 'x1' – 'x10' |
| defaults | log-transform | 'x1' |
| | | 'x18' – 'x23' |
| | min-max scaling | 'x12' – 'x17' |
| house | log-transform | 'P1' |
| houses | log-transform | 'total rooms' |
| | | 'total bedrooms' |
| | | 'population' |
| | | 'households' |
| higgs | min-max scaling | 'lepton_pT' |
| | | 'lepton_eta' |
| | | 'lepton_phi' |
| | | 'missing_energy_magnitude' |
| | | 'missing_energy_phi' |
| | | 'jet_1_pt' – 'jet_4_pt' |
| | | 'jet_1_eta' – 'jet_4_eta' |
| | | 'jet_1_phi' – 'jet_4_phi' |
| | | 'm_jj' |
| | | 'm_jjj' |
| | | 'm_lv' |
| | | 'm_jlv' |
| | | 'm_bb' |
| | | 'm_wbb' |
| | | 'm_wwbb' |
| medical_charges | log-transform | 'Average Covered Charges' |
| | | 'Average Medicare Payments' |
| miniboone | min-max scaling | 'ParticleID 19' |
| nyc_taxi | min-max scaling | 'passenger_count' |
| | | 'tolls_amount' |
| | | 'total_amount' |
| | | 'tip_amount' |
| road_safety | min-max scaling | 'Location_Easting_OSGR' |
| | | 'Location_Northing_OSGR' |
| | log-transform | 'Engine_Capacity_(CC)' |
| soil | min-max scaling | 'northing' |
| | | 'easting' |
| | | 'resistivity' |
| | | 'track' |

Supplmentary Table 1: The numerical variables subjected to min-max scaling, or log-transformation.

| | Method | | | |
|---|---|---|---|---|
| **Dataset** | CLIP | GMC | MCN | **ICE-T** |
| abalone | 0.983 | 0.983 | 0.982 | 0.999 |
| ailerons | 0.797 | 0.887 | 0.732 | 0.844 |
| airlines | 0.814 | 0.782 | 0.838 | 0.869 |
| albert | | 0.396 | | 0.648 |
| allstate | | 0.217 | | 0.735 |
| bank_marketing | 0.722 | 0.974 | 0.900 | 0.755 |
| bike_sharing | 0.857 | 0.889 | 0.835 | 0.846 |
| bioresponse | | 0.989 | | 0.984 |
| brazilian_houses | 0.889 | 0.769 | 0.845 | 0.789 |
| california | 0.923 | 0.852 | 0.722 | 0.848 |
| clothing | 0.729 | 0.416 | 0.801 | 0.772 |
| compas | 0.966 | 0.596 | 0.960 | 0.937 |
| covertype | 0.937 | 0.955 | 0.887 | 0.868 |
| cpu | 0.954 | 0.920 | 0.923 | 0.891 |
| credit | 0.831 | 0.968 | 0.982 | 0.942 |
| defaults | 0.920 | 0.881 | 0.857 | 0.887 |
| diabetes | 0.751 | 0.978 | 0.866 | 0.998 |
| diamonds | 0.802 | 0.812 | 0.797 | 0.862 |
| electricity | 0.942 | 0.879 | 0.836 | 0.949 |
| elevators | 0.868 | 0.946 | 0.674 | 0.895 |
| eye_movements | 0.801 | 0.830 | 0.808 | 0.795 |
| heloc | 0.926 | 0.946 | 0.957 | 0.934 |
| higgs | | 0.918 | | 0.940 |
| house | 0.971 | 0.963 | 0.959 | 0.981 |
| house_sales | 0.886 | 0.876 | 0.674 | 0.810 |
| houses | 0.884 | 0.879 | 0.862 | 0.850 |
| jannis | | 0.927 | | 0.780 |
| kickstarter | 0.890 | 0.551 | 0.981 | 0.939 |
| medical_charges | 0.918 | 0.999 | 0.979 | 0.999 |
| mercedes | | 0.046 | | 0.730 |
| miami_housing | 0.655 | 0.933 | 0.753 | 0.887 |
| miniboone | | 0.956 | | 0.930 |
| nyc_taxi | 0.862 | 0.485 | 0.856 | 0.734 |
| pol | 0.872 | 0.963 | 0.973 | 0.977 |
| road_safety | | 0.430 | | 0.743 |
| seattle_crime | 0.984 | 0.936 | 0.982 | 1.000 |
| sgemm | 0.803 | 0.891 | 0.677 | 0.846 |
| skin_cancer | 0.992 | 0.773 | 0.504 | 0.986 |
| soil | 0.857 | 0.759 | 0.800 | 0.786 |
| streetview | 0.852 | 1.000 | 0.914 | 0.931 |
| sulfur | 0.950 | 0.877 | 0.937 | 0.998 |
| superconduct | | 0.855 | | 0.818 |
| supreme | 0.818 | 0.607 | 0.578 | 0.576 |
| telescope | 0.922 | 0.970 | 0.902 | 0.990 |
| topo | | 0.977 | | 0.895 |
| ukair | 0.766 | 0.643 | 0.628 | 0.684 |
| wine_quality | 0.916 | 0.938 | 0.867 | 0.967 |
| yprop | | 0.970 | | 0.961 |

Supplmentary Table 2: Imputation

| | Method | | | | | | |
|---|---|---|---|---|---|---|---|
| **Dataset** | Benchmark | Scarf | SubTab | CLIP | GMC | MCN | **ICE-T** |
| *Silhouette score - higher is better* | | | | | | | |
| abalone | 0.640 | 0.934 | 0.872 | 0.862 | 0.688 | 0.828 | 0.618 |
| ailerons | 0.518 | 0.749 | 0.569 | 0.676 | 0.603 | 0.570 | 0.730 |
| airlines | 0.242 | 0.851 | 0.441 | 0.358 | 0.888 | 0.280 | 0.742 |
| albert | 0.024 | 0.387 | 0.284 | | 0.252 | | 0.413 |
| allstate | 0.074 | 0.599 | 0.639 | | 0.619 | | 0.575 |
| bank_marketing | 0.831 | 0.916 | 0.827 | 0.830 | 0.806 | 0.824 | 0.916 |
| bike_sharing | 0.418 | 0.848 | 0.591 | 0.476 | 0.367 | 0.415 | 0.753 |
| bioresponse | 0.104 | 0.552 | 0.500 | | 0.603 | | 0.671 |
| brazilian_houses | 0.352 | 0.482 | 0.391 | 0.404 | 0.680 | 0.352 | 0.557 |
| california | 0.703 | 0.889 | 0.711 | 0.730 | 0.987 | 0.716 | 0.982 |
| clothing | | 0.773 | 0.554 | 0.750 | 0.891 | 0.476 | 0.909 |
| compas | 0.555 | 0.761 | 0.566 | 0.521 | 0.546 | 0.686 | 0.812 |
| covertype | 0.329 | 0.863 | 0.490 | 0.319 | 0.725 | 0.339 | 0.687 |
| cpu | 0.379 | 0.473 | 0.571 | 0.437 | 0.642 | 0.429 | 0.343 |
| credit | 0.734 | 0.997 | 0.906 | 0.905 | 0.998 | 0.904 | 0.931 |
| defaults | 0.318 | 0.574 | 0.446 | 0.519 | 0.704 | 0.401 | 0.760 |
| diabetes | 0.429 | 0.644 | 0.554 | 0.576 | 0.253 | 0.459 | 0.463 |
| diamonds | 0.231 | 0.631 | 0.401 | 0.355 | 0.690 | 0.596 | 0.744 |
| electricity | 0.576 | 0.564 | 0.833 | 0.854 | 0.628 | 0.677 | 0.998 |
| elevators | 0.556 | 0.584 | 0.596 | 0.569 | 0.477 | 0.567 | 0.577 |
| eye_movements | 0.553 | 0.914 | 0.792 | 0.910 | 0.920 | 0.756 | 0.885 |
| heloc | 0.289 | 0.342 | 0.437 | 0.422 | 0.504 | 0.971 | 0.288 |
| higgs | 0.115 | 0.842 | 0.686 | | 0.858 | | 0.476 |
| house | 0.490 | 0.760 | 0.819 | 0.546 | 0.591 | 0.499 | 0.649 |
| house_sales | 0.947 | 0.957 | 0.945 | 0.947 | 0.947 | 0.946 | 0.927 |
| houses | 0.525 | 0.772 | 0.613 | 0.580 | 0.803 | 0.558 | 0.846 |
| jannis | 0.227 | 0.391 | 0.361 | | 0.788 | | 0.505 |
| kickstarter | | 0.987 | 0.994 | 0.935 | 0.788 | 0.985 | 0.879 |
| medical_charges | 0.800 | 0.959 | 0.800 | 0.805 | 0.926 | 0.818 | 0.801 |
| mercedes | 0.135 | 0.474 | 0.187 | | 0.771 | | 0.554 |
| miami_housing | 0.347 | 0.761 | 0.430 | 0.410 | 0.815 | 0.786 | 0.876 |
| miniboone | 0.646 | 0.994 | 1.000 | | 0.996 | | 1.000 |
| nyc_taxi | 0.090 | 0.524 | 0.470 | 0.266 | 0.795 | 0.201 | 0.772 |
| pol | 0.173 | 0.333 | 0.417 | 0.305 | 0.581 | 0.359 | 0.629 |
| road_safety | 0.297 | 0.358 | 0.596 | | 0.937 | | 0.392 |
| seattle_crime | 0.185 | 0.326 | 0.208 | 0.267 | 0.212 | 0.211 | 0.442 |
| sgemm | 0.828 | 0.859 | 0.835 | 0.839 | 0.912 | 0.835 | 0.888 |
| skin_cancer | | 0.969 | 0.971 | 0.988 | 0.849 | 0.993 | 0.849 |
| soil | 0.635 | 0.994 | 0.682 | 0.906 | 0.703 | 0.925 | 0.681 |
| streetview | | 0.819 | 0.959 | 0.987 | 0.738 | 0.745 | 0.851 |
| sulfur | 0.489 | 0.620 | 0.533 | 0.545 | 0.738 | 0.607 | 0.754 |
| superconduct | 0.535 | 0.739 | 0.620 | | 0.678 | | 0.621 |
| supreme | 0.575 | 0.920 | 0.523 | 0.621 | 0.628 | 0.577 | 0.841 |
| telescope | 0.435 | 0.732 | 0.640 | 0.474 | 0.763 | 0.640 | 0.729 |
| topo | 0.304 | 0.864 | 0.496 | | 0.558 | | 0.813 |
| ukair | 0.701 | 0.640 | 0.831 | 0.873 | 0.107 | 0.793 | 0.968 |
| wine_quality | 0.510 | 0.691 | 0.576 | 0.526 | 0.576 | 0.526 | 0.556 |
| yprop | 0.288 | 0.478 | 0.343 | | 0.638 | | 0.531 |

Supplmentary Table 3: Clustering

| | Method | | | | | | |
|---|---|---|---|---|---|---|---|
| **Dataset** | Benchmark | Scarf | SubTab | CLIP | GMC | MCN | **ICE-T** |
| *Balanced accuracy score (Classification) – higher is better* | | | | | | | |
| albert | 59.307 | 61.819 | 60.814 | | 60.342 | | 61.622 |
| bank_marketing | 75.664 | 76.729 | 75.497 | 75.409 | 77.277 | 75.792 | 76.276 |
| bioresponse | 72.862 | 69.453 | 73.011 | | 66.829 | | 72.173 |
| california | 62.433 | 67.873 | 66.104 | 82.291 | 80.639 | 77.441 | 78.223 |
| clothing | | 40.803 | 40.202 | 42.743 | 39.700 | 40.375 | 46.294 |
| compas | 65.251 | 66.378 | 65.712 | 66.600 | 66.141 | 66.087 | 66.573 |
| covertype | 81.749 | 64.767 | 61.607 | 81.292 | 71.793 | 72.935 | 68.549 |
| credit | 56.977 | 73.162 | 67.234 | 61.209 | 76.824 | 62.900 | 73.311 |
| defaults | 66.983 | 70.741 | 71.138 | 68.380 | 69.777 | 70.743 | 70.205 |
| diabetes | 54.675 | 57.978 | 56.188 | 57.550 | 57.756 | 57.027 | 57.608 |
| electricity | 77.509 | 78.768 | 82.393 | 82.958 | 80.631 | 82.424 | 83.330 |
| eye_movements | 55.701 | 56.748 | 59.604 | 60.150 | 58.434 | 58.805 | 56.119 |
| heloc | 68.189 | 67.679 | 68.853 | 68.602 | 68.433 | 68.738 | 68.888 |
| higgs | 53.092 | 53.085 | 53.350 | | 58.309 | | 64.890 |
| jannis | 64.003 | 72.804 | 69.209 | | 70.338 | | 73.152 |
| kickstarter | | 53.337 | 53.763 | 57.094 | 56.819 | 56.716 | 56.457 |
| miniboone | 87.114 | 89.863 | 89.041 | | 89.521 | | 91.398 |
| road_safety | 58.925 | 72.709 | 70.714 | | 70.670 | | 64.671 |
| skin_cancer | | 61.266 | 65.955 | 64.830 | 65.062 | 63.886 | 60.173 |
| streetview | | 87.954 | 90.913 | 90.062 | 87.260 | 90.789 | 89.480 |
| telescope | 77.112 | 77.287 | 78.645 | 78.934 | 77.969 | 80.584 | 81.035 |
| *MSE (Regression) – lower is better* | | | | | | | |
| abalone | 4.69957 | 5.31256 | 4.32163 | 4.37282 | 5.95043 | 4.72089 | 5.38935 |
| ailerons | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| airlines | 0.13671 | 0.01859 | 0.01871 | 0.01859 | 0.01860 | 0.01852 | 0.01865 |
| allstate | 0.05829 | 0.00331 | 0.00321 | | 0.00344 | | 0.00317 |
| bike_sharing | 21032.67723 | 10434.51400 | 12173.03800 | 12643.51400 | 10324.40200 | 10863.28400 | 11613.92900 |
| brazilian_houses | 0.00038 | 0.00051 | 0.00103 | 0.00013 | 0.00031 | 0.00033 | 0.00007 |
| cpu | 9.52330 | 8.45528 | 7.63315 | 7.82178 | 10.76382 | 7.83379 | 8.98616 |
| diamonds | 0.04037 | 0.02019 | 0.02396 | 0.03312 | 0.01756 | 0.02049 | 0.03846 |
| elevators | 0.00004 | 0.00004 | 0.00004 | 0.00004 | 0.00004 | 0.00003 | 0.00003 |
| house | 0.42876 | 0.53404 | 0.43986 | 0.47898 | 0.50120 | 0.44088 | 0.42927 |
| house_sales | 0.12313 | 0.11423 | 0.11395 | 0.11756 | 0.11229 | 0.11384 | 0.10987 |
| houses | 0.09530 | 0.15153 | 0.07202 | 0.08437 | 0.14501 | 0.08823 | 0.12270 |
| medical_charges | 0.02028 | 0.02144 | 0.01616 | 0.02122 | 0.01109 | 0.01232 | 0.01241 |
| mercedes | 75.09708 | 59.27377 | 43.83995 | | 70.46438 | | 48.39973 |
| miami_housing | 0.05099 | 0.03833 | 0.04331 | 0.04673 | 0.08245 | 0.03124 | 0.04209 |
| nyc_taxi | 0.06974 | 0.00572 | 0.00520 | 0.00568 | 0.00400 | 0.00554 | 0.00403 |
| pol | 60.49084 | 57.50370 | 50.97037 | 64.06667 | 35.07755 | 47.69167 | 65.23950 |
| seattle_crime | 160450.58896 | 156052.69000 | 156762.48000 | 157684.94000 | 157440.58000 | 155453.16000 | 157572.48000 |
| sgemm | 0.00024 | 0.00027 | 0.00027 | 0.00027 | 0.00027 | 0.00031 | 0.00026 |
| soil | 0.00598 | 0.00000 | 0.00016 | 0.00003 | 0.00003 | 0.00004 | 0.00008 |
| sulfur | 0.00037 | 0.00022 | 0.00022 | 0.00018 | 0.00020 | 0.00035 | 0.00022 |
| superconduct | 134.12197 | 130.40073 | 108.44905 | | 104.98537 | | 102.89780 |
| supreme | 0.00735 | 0.00633 | 0.00714 | 0.00495 | 0.00536 | 0.00657 | 0.00452 |
| topo | 0.00080 | 0.00082 | 0.00081 | | 0.00082 | | 0.00082 |
| ukair | 0.15621 | 0.16689 | 0.15702 | 0.16710 | 0.19882 | 0.18213 | 0.15919 |
| wine_quality | 0.64618 | 0.56382 | 0.57846 | 0.54574 | 0.62226 | 0.53675 | 0.51462 |
| yprop | 0.00078 | 0.00078 | 0.00074 | | 0.00080 | | 0.00077 |

Supplmentary Table 4: Supervised learning

| **Dataset** | **Method** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Benchmark | Control | Scarf | SubTab | CLIP | GMC | MCN | **ICE-T** |
| *Classification (ACC) – higher is better* | | | | | | | | |
| albert | 65.047 | 65.181 | 65.560 | 65.379 | | 65.550 | | 65.584 |
| bank_marketing | 78.321 | 78.456 | 78.579 | 77.982 | 77.158 | 80.357 | 78.698 | 78.814 |
| bioresponse | 77.677 | 75.920 | 77.949 | 77.354 | | 77.002 | | 79.221 |
| california | 90.584 | 80.932 | 80.977 | 81.108 | 80.734 | 80.745 | 80.636 | 81.120 |
| clothing | | 37.933 | 38.452 | 37.999 | 41.178 | 36.518 | 38.673 | 41.463 |
| compas | 66.789 | 68.589 | 68.838 | 68.992 | 68.296 | 68.382 | 68.463 | 68.742 |
| covertype | 85.289 | 87.615 | 92.624 | 77.242 | 88.715 | 92.715 | 88.649 | 92.806 |
| credit | 76.320 | 77.423 | 76.900 | 77.113 | 76.914 | 77.566 | 77.184 | 77.852 |
| defaults | 69.842 | 72.477 | 72.548 | 72.643 | 72.610 | 72.953 | 72.494 | 73.031 |
| diabetes | 60.404 | 61.024 | 60.931 | 60.991 | 60.998 | 60.943 | 60.848 | 61.146 |
| electricity | 90.827 | 81.659 | 81.807 | 82.873 | 83.039 | 84.259 | 82.416 | 83.893 |
| eye_movements | 66.645 | 56.760 | 55.920 | 56.078 | 56.279 | 57.923 | 56.544 | 56.799 |
| heloc | 70.902 | 69.512 | 70.944 | 69.955 | 70.279 | 70.234 | 69.930 | 71.108 |
| higgs | 73.538 | 73.706 | 74.985 | 72.168 | | 74.447 | | 74.796 |
| jannis | 78.862 | 78.066 | 79.597 | 77.164 | | 79.143 | | 79.855 |
| kickstarter | | 63.774 | 62.200 | 60.216 | 62.569 | 63.055 | 66.431 | 64.686 |
| miniboone | 93.855 | 93.905 | 93.918 | 94.070 | | 94.535 | | 94.672 |
| road_safety | 78.225 | 77.763 | 79.140 | 77.843 | | 78.534 | | 79.690 |
| skin_cancer | | 96.487 | 95.404 | 96.468 | 95.977 | 95.135 | 96.186 | 94.748 |
| streetview | | 59.338 | 80.257 | 67.441 | 69.276 | 70.629 | 71.825 | 78.739 |
| telescope | 85.972 | 84.020 | 83.357 | 83.351 | 85.072 | 85.737 | 83.809 | 86.768 |
| *Regression (MSE) – lower is better* | | | | | | | | |
| abalone | 5.05978 | 3.91145 | 4.05962 | 3.91939 | 3.86525 | 4.01733 | 3.85508 | 3.96275 |
| ailerons | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 | 0.00000 |
| airlines | 0.12965 | 0.01696 | 0.01693 | 0.01709 | 0.01693 | 0.01694 | 0.01693 | 0.01691 |
| allstate | 0.04847 | 0.00247 | 0.00242 | 0.00264 | | 0.00242 | | 0.00243 |
| bike_sharing | 10929.90035 | 9875.94700 | 9741.88400 | 9787.36900 | 9651.51000 | 9737.07600 | 9845.23000 | 9862.71600 |
| brazilian_houses | 0.00003 | 0.00008 | 0.00007 | 0.00007 | 0.00003 | 0.00004 | 0.00002 | 0.00004 |
| cpu | 6.33514 | 8.69058 | 8.53732 | 8.18825 | 6.96726 | 7.25649 | 6.67351 | 6.30104 |
| diamonds | 0.00830 | 0.01619 | 0.01366 | 0.01460 | 0.01511 | 0.01269 | 0.01467 | 0.01093 |
| elevators | 0.00001 | 0.00004 | 0.00005 | 0.00004 | 0.00004 | 0.00004 | 0.00004 | 0.00003 |
| house | 0.37756 | 0.47359 | 0.47892 | 0.45266 | 0.44590 | 0.48754 | 0.43751 | 0.43130 |
| house_sales | 0.03019 | 0.15205 | 0.08681 | 0.19229 | 0.15862 | 0.13981 | 0.12547 | 0.16002 |
| houses | 0.05121 | 0.11072 | 0.10974 | 0.10562 | 0.10344 | 0.10236 | 0.11376 | 0.10352 |
| medical_charges | 0.00779 | 0.00684 | 0.00699 | 0.00701 | 0.00693 | 0.00684 | 0.00683 | 0.00687 |
| mercedes | 69.05531 | 42.33849 | 43.48200 | 43.08921 | | 43.74935 | | 44.94259 |
| miami_housing | 0.02473 | 0.09113 | 1.01151 | 0.12189 | 0.22357 | 0.11804 | 0.09870 | 0.11192 |
| nyc_taxi | 0.05003 | 0.00272 | 0.00275 | 0.00363 | 0.00276 | 0.00336 | 0.00269 | 0.00253 |
| pol | 26.52296 | 38.61242 | 32.18196 | 37.99794 | 34.84072 | 45.15300 | 38.72781 | 26.15045 |
| seattle_crime | 147748.56894 | 146966.23000 | 147000.86000 | 146991.89000 | 146976.47000 | 147479.90000 | 147045.56000 | 147406.19000 |
| sgemm | 0.00029 | 0.00063 | 0.00053 | 0.00066 | 0.00037 | 0.00028 | 0.00073 | 0.00029 |
| soil | 0.00151 | 0.00001 | 0.00001 | 0.00081 | 0.00001 | 0.00001 | 0.00001 | 0.00001 |
| sulfur | 0.00091 | 0.00129 | 0.00101 | 0.00119 | 0.00104 | 0.00108 | 0.00082 | 0.00090 |
| superconduct | 98.50537 | 256.64734 | 246.62310 | 251.30844 | | 229.29940 | | 230.40335 |
| supreme | 0.00756 | 0.29500 | 0.29459 | 0.29417 | 0.29533 | 0.29988 | 0.29496 | 0.29220 |
| topo | 0.00083 | 0.00075 | 0.00075 | 0.00075 | | 0.00077 | | 0.00075 |
| ukair | 0.13846 | 0.14539 | 0.14442 | 0.14684 | 0.14124 | 0.13942 | 0.14231 | 0.12723 |
| wine_quality | 0.51409 | 0.51571 | 0.51537 | 0.51922 | 0.50477 | 0.52282 | 0.51265 | 0.51159 |
| yprop | 0.00086 | 0.00076 | 0.00077 | 0.00076 | | 0.00076 | | 0.00075 |

Supplmentary Table 5: Transfer learning

# 4 Statistical analysis

We tested the statistical significance of the obtained results under the alternative hypothesis that the performance obtained by the ICE-T across the 48 datasets, is greater than the one obtained by the baseline methods collectively, using the one sided T-test. The obtained $p$-values (lower the better) are summarized in the table below. The results show that ICE-T outperforms the baselines very significantly ($p < 0.01$) in three out of four tasks and with lower significance ($p = 0.065$) in one of the tasks.

| **Task** | **p-value** | **Statistic** |
|---|---|---|
| Imputation | 6.506e-02 | 1.528 |
| Clustering | 6.488e-03 | 2.552 |
| Supervised learning | 3.003e-03 | 2.832 |
| Transfer learning | 4.415e-10 | 6.921 |

Supplmentary Table 6: Results of the statistical evaluation.

# 5 Scalability statement

We would like to emphasize that certain extremely high-dimensional tabular datasets may remain computationally challenging even despite ICE-T's favorable scaling. However, this limitation reflects a general trade-off for all CRL methods, which employ column, or modality-specific embeddings. We emphasize that ICE-T was successfully tested on datasets with several hundred columns and up to 1 million rows (Table 3 in the manuscript), which we believe is representative of the majority of real-world tabular datasets commonly encountered in practice.

# References


[1] Vincent Cohen-Addad et al. "Online k-means clustering". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2021, pp. 1126–1134.

[2] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[3] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.