NTU Machine Learning Final Project Proposal Notes

tags: `NTU_ML` `Machine Learning`

Deep6mAPred: A CNN and Bi-LSTM-based deep learning method for predicting DNA N6-methyladenosine sites across plant species

For this paper, my perspective is this is a little bit trivial to solve the problem. For simplicity speaking, they just change the stacking model structure to a sequence structure. In addition, the result of this paper is exaggerating.

Comparison

- The result below is the experience on `6mA-rice-LV`(rice) dataset, and this paper method is `Deep6mAPred`. They used 5-fold cross validation on this data(`6mA-rice-LV`). In the original context, they said:

  > The `Deep6mAPred` reached better Sn than three baseline methods (`Deep6mA`, `SNNRice6mA-large` and `Deep6mAPred`), and achieved competitive SP, ACC and MCC in contrast with the `Deep6mA`, which completely outperformed the `SNNRice6mA-large` and MM-6mAPred.

  However, the fun fact is the performance of Sp, ACC, MCC, AUC is not good enough in this dataset.

  Performances by 5-fold cross validation over the 6mA-rice-Lv.

Method	S_n	S_p	ACC	MCC	AUC
Deep6mAPred	**0.9538**	0.9255	0.9397	0.8798	0.9793
Deep6mA*	0.9506	**0.9296**	**0.9401**	**0.8800**	**0.9800**
SNNRice6mA-large*	0.9347	0.8975	0.9204	0.8400	0.9700
MM-6mApred *	0.9347	0.8951	0.9149	0.8300	0.9600

  The asterisk (*) indicated that the results were from the literature [83].

- The result below is for `6mA-rice-chen` dataset. Compared with `Deep6mA`, `Deep6mAPred` increased Sn by 0.1572, ACC by 0.0750, MCC by 0.1436, and AUC of ROC curve by 0.0237, completely superior to the other two methods. The Sp of `Deep6mAPred` is slightly lower than that of `Deep6mA`, but much higher than that of the other two methods.

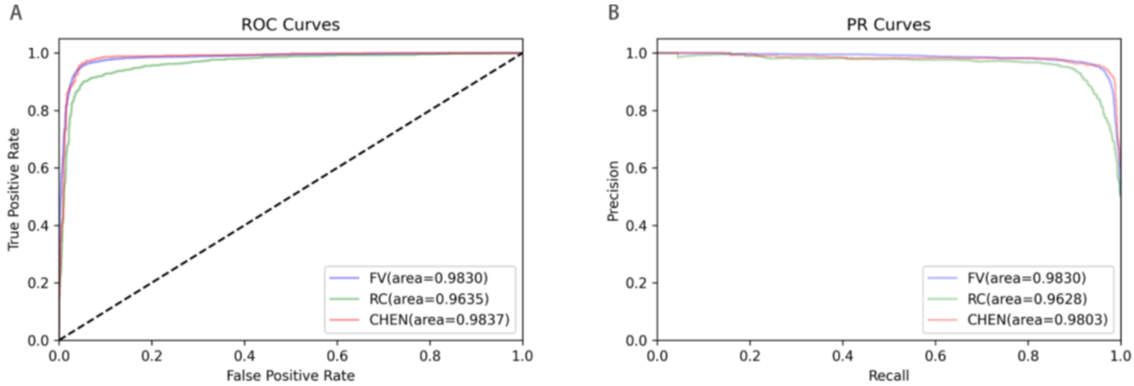
  Performances over the 6mA-rice-chen dataset.

Method	S_n	S_p	ACC	MCC	AUC
Deep6mAPred	**0.9545**	0.9568	**0.9556**	**0.9136**	**0.9837**
Deep6mA*	0.7973	**0.9640**	0.8806	0.7700	0.9600
SNNRice6mAlarge*	0.7790	0.8742	0.8267	0.6500	0.8900
MM-6mApred*	0.7682	0.9170	0.8426	0.6800	0.9100

  The asterisk (*) indicated that the results were from the literature [83].

  This result is quite distinguished that can show how special their model is under this another rice data.

- This is ROC curves and PR curves result on `6mA-Fuse-R` (Rosa chinensis) and `6mA-Fuse-F` (Fragaria vesca, a kind of wild strawberry) respectively. In order to show how robust on their model, they try to test different species such as rose and wild strawberry without training, and the result is quite significant that almost similar to rice data.



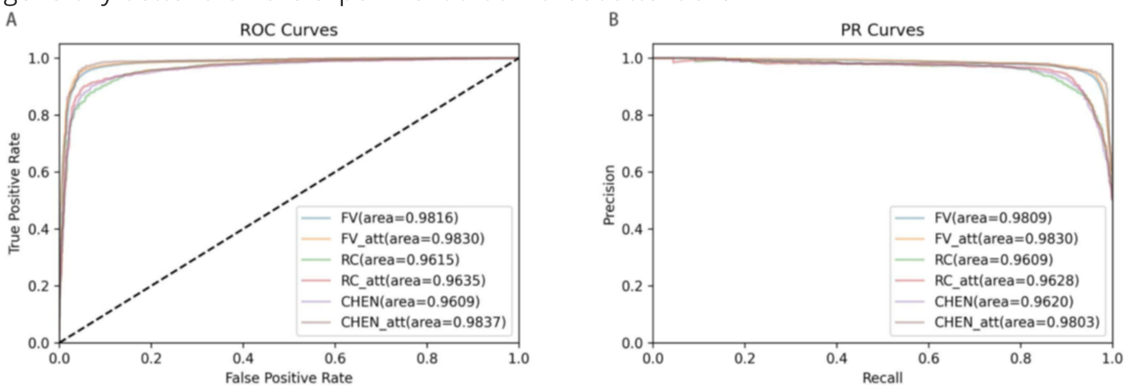
- This is a self-created table that I wanna show the AUC of two curves with different species. The original context said:

> As for the 6mA-Fuse-R, the `Deep6mAPred` outperformed three baseline methods in terms of the AUCs of ROC curves, while in terms of the AUCs of the PR curves it was equivalent to the `Deep6mA` but superior to the `SNNRice6mA-large` and `MM-6mAPred` a bit

Follow the description above, we can know that the result of `6mA-Fuse-R` is better than three baseline methods but without any table or figure to prove that and this is not rigourous enough for this information.

Model	AUC of ROC Curves		AUC of PR Curves	
	6mA-Fuse-F	6mA-Fuse-R	6mA-Fuse-F	6mA-Fuse-R
Deep6mAPred	0.9830	0.9635	0.9830	0.9628
Deep6mA*	0.9820	N/A	0.9820	N/A
SNNRice6mA-large*	0.9640		0.9630	
MM-6mApred*	0.9600		0.9590	

- They also do some ablation experiment to prove that the attention mechanism they choose is quite valid and useful in this project.
We can see that in each experiment of different species, with attention mechanism is generally better than the experiment that without attention.



Other Issue

- Why can wild rose and rice use the same architecture or we can ask how to process input data so that they can be applicable at the same model structure.
- There is no extra explanation for the selected attention mechanism method.

Ensemble Learning for Brain Age Prediction

The main opinion to this paper is that it's report of the competition they attended. And listed as clear as possible what problems they encountered, what techniques they used etc.

Comparison

- The * symbol represents a significant reduction in MAE by Ensemble Learning compared to Inception alone ($p\ value < 0.05$)
  - For the objective of minimize MAE, the way of deep learning is better than `BLUP` and `SVM` ($pvalue\ of\ paired\ t-test<3.1e-4$)
  - There was no significant difference in the performance of the deep learning algorithms ($p > 0.027$)
  - In contrast, Ensemble Learning's $MAE=3.46$, there is a significant difference (p=1.3e-4)
  - Taking challenge 2 as an example, the author uses median and mean absolute deviation per site to rescale the prediction. The results show that MAE will increase by one year compared to the original one, but will reduce the bias. The same that ensemble learning has a significant improvement compared to Inception($p=0.010$).

		BLUP-mean	BLUP-quantiles	SVM	6-layer CNN	Age spe. 6-layer CNN	ResNet	Inception V1	Ensemble prediction	PAC results
First challenge	MAE (SE)	5.32 (0.19)	4.90 (0.19)	5.31 (0.18)	4.18 (0.16)	4.01 (0.15)	4.02 (0.15)	3.82 (0.14)	3.46 (0.13)*	3.33
	$\|\rho\|$	0.32	0.37	0.58	0.25	0.30	0.24	0.41	0.32	0.21
Second challenge	MAE (SE)	6.15 (0.23)	5.96 (0.23)	6.14 (0.23)	5.27 (0.21)	5.17 (0.20)	5.25 (0.20)	4.97 (0.19)	4.69 (0.19)*	4.83
	$\|\rho\|$	0.14	0.15	0.15	0.084	0.068	0.11	0.058	0.058	0.021

- They also tried to evaluate whether their conclusions depend on the train/test split used in the previous section by performing a 5-fold cross-validation experiment.
  - Within each fold, they found a nominally significant difference in MAE between `BLUP`/`SVM` and `ResNet` ($p < 5.5E−3$)
  - In each fold, the composite age score using linear regression outperformed `Inception V1`'s predictions ($p < 0.0022$). For folds 2 and 3, ensemble learning via random trees significantly outperforms `Inception V1` alone ($p=4.0E−3 and 3.4E−4$)
  - Note that the MAE obtained using Random Forest is very close to the MAE obtained by taking the mean or median score for each person. We cannot conclude that there is a significant difference between **linear model combinations** and **random forests**.

	Individual algorithms							Ensemble learning			
	BLUP-mean	BLUP-quantiles	SVM	6-layer CNN	Age spe. 6-layer CNN	ResNet	Inception V1	LM	RF	Mean	Median
Fold 1	5.32 (0.19)	4.90 (0.19)	5.31 (0.18)	4.18 (0.16)	4.01 (0.15)	4.02 (0.15)	3.82 (0.14)	3.46 (0.13)*	3.62 (0.15)	3.74 (0.13)	3.67 (0.14)
Fold 2	5.05 (0.18)	4.79 (0.19)	5.34 (0.18)	4.47 (0.15)	4.12 (0.13)	4.01 (0.14)	3.97 (0.15)	3.53 (0.13)*	3.60 (0.15)*	3.69 (0.13)	3.74 (0.13)
Fold 3	4.90 (0.18)	4.37 (0.16)	4.84 (0.17)	4.41 (0.16)	4.27 (0.15)	3.88 (0.14)	4.00 (0.16)	3.33 (0.13)*	3.46 (0.15)*	3.46 (0.12)*	3.45 (0.13)*
Fold 4	5.07 (0.18)	4.71 (0.18)	5.06 (0.18)	4.55 (0.17)	4.27 (0.16)	4.11 (0.15)	3.85 (0.15)	3.57 (0.13)*	3.72 (0.14)	3.68 (0.14)	3.74 (0.15)
Fold 5	5.22 (0.19)	4.69 (0.18)	5.20 (0.18)	4.02 (0.16)	3.89 (0.15)	3.99 (0.16)	3.75 (0.15)	3.34 (0.13)*	3.51 (0.14)	3.56 (0.13)	3.47 (0.13)
5-fold combined MAE	5.11	4.69	5.15	4.33	4.11	4.00	3.88	3.44	3.58	3.62	3.61

Significant Difference ?

- The low performance of `BLUP` / `SVM` shown above compared to deep learning algorithms motivated the authors to test whether it could be attributed to the input data or the algorithm itself. Therefore, the author retrains `BLUP` and `SVM` (trained on gray matter maps)
  - † Symbols represent: the algorithm trained with gray matter map is significantly **better than** the algorithm trained with surface-based vertices ($p < 0.05/15$).
  - The * symbol indicates: the performance of the algorithm trained on the gray matter image is significantly **lower than** that of `Inception V1` ($p < 0.05/15$)
  - Despite the reduction in MAE, `BLUP-mean` and `SVM` trained on gray matter still performed **worse than** `Inception V1` ($p < 0.0033$), although the difference between `Inception V1` and `BLUP-quantile` became not significant.

	BLUP-mean	BLUP-quantiles	SVM	Ensemble learning
Fold 1	4.51 (0.16)$^{\dagger *}$	3.91 (0.14)†	4.64 (0.17)$^{\dagger *}$	3.39 (0.13)
Fold 2	4.45 (0.16)$^{\dagger *}$	4.06 (0.15)†	4.75 (0.16)$^{\dagger *}$	3.46 (0.13)
Fold 3	4.67 (0.17)*	4.02 (0.16)	4.62 (0.17)*	3.26 (0.13)
Fold 4	4.59 (0.16)*	4.16 (0.16)†	4.52 (0.16)*	3.55 (0.14)
Fold 5	4.86 (0.18)*	4.21 (0.17)	4.78 (0.17)*	3.35 (0.14)
5-fold MAE	4.61	4.07	4.66	3.42

- The participant is older, the prediction error is larger. → Therefore, the predictor will tends to underestimate the age of older participants and overestimate the age of younger participants.
  We did not observe significant associations of prediction errors with gender or location

	BLUP-mean	BLUP-quantiles	SVM	6-layer CNN	Age spe. 6-layer CNN	ResNet	Inception V1
Age	2.9E−10*	5.8E−13*	5.8E−46*	7.3E−10*	2.2E−13*	9.1E−05*	7.7E−20*
Site	3.7E−01	4.4E−02	4.5E−03	2.8E−02	4.3E−02	2.3E−02	5.0E−02
Sex	7.1E−02	1.4E−01	3.6E−02	1.0E+00	8.5E−01	1.0E+00	5.4E−01

Other Issue

- They didn't explain why they used two `6-Layers CNN` to combine and the effect in detailed.
- They also didn't explain the gray/white matter map difference and the properties of these maps in detailed.

Machine learning workflows

This paper is just like a `Readme` file that wanna teach someone how to use their tool, each technique they used, each problem they encountered, and also which programming package they used etc. as clear as possible. Although the paper should be as clear as possible, but too much unnecessary information is really a waste of time and annoying.

Comparison

- **Random Forest(RFs)**
  - Vanilla RF(vRF)
    - The ME of vRF was 4.8%, the AUC was 99.9%, and the corresponding BS and LL were 0.32 and 0.78, respectively
    - Platt scaling with LR and `FLR` improves BS and LL by a factor of 2-4, furthermore, `FLR` is better than `LR`

        - MR slightly outperformed Platt's two variants and achieved very low 10th and 9th overall BS (0.073) and LL (0.155) metrics respectively
    - tuned RF(tRF)
        - RF tuned for ME (`tRFME`) showed 10th overall error rate (3.5%) and 4th AUC (99.9%), while it had relatively high BS (0.35) and LL (0.86) similar to `vRF`
        - Both `tRFBS` and `tRFLL` have higher error rates, about 5.5%
        - After calibration with **MR**, almost all versions of `tRF` get the biggest performance improvement

Workflow	Top 10 BS	Classifier	Run-time 5 × 5 CV (/fold)	No. of CPU threads [hardware]	R package	Calibrator	Optimized metric	Hyperparameters	ME	AUC	BS	LL
vRF		RF	38 min	1 [2]	randomForest	raw	ME	ntree =500, mtry=100	0.048	0.999	0.320	0.780
vRF + LR		RF	+30 s (LR)	1 [2]	randomForest	Platt LR (us)	ME	p_{varsel} = 200	0.052	-	0.106	0.289
vRF + LR		RF	+30 s (LR)	1 [2]	randomForest	Platt LR	ME	p_{varsel} = 200	0.052	0.994	0.081	0.262
vRF + FLR		RF	+8-9 min (FLR)	1 [2]	randomForest	Platt Firth (us)	ME	p_{varsel} = 200	0.048	-	0.105	0.193
vRF + FLR		RF	+8-9 min (FLR)	1 [2]	randomForest	Platt Firth	ME	p_{varsel} = 200	0.048	0.999	0.081	0.193
vRF + MR	10	RF	+7--8 min (MR)	11 [2]	randomForest	MR	ME	p_{varsel} = 200	0.043	0.999	0.073	0.155
tRF_{BS}		RF	12-13 h (16-25 min/fold)	72 [5]	randomForest	raw	BS	ntree = (500, 1,000, 1,500, 2,000)	0.055	0.999	0.272	0.673
tRF_{ME}		RF			randomForest	raw	ME	mtry = (80, 90, 100, 110)	0.035	0.999	0.351	0.855
tRF_{LL}		RF			randomForest	raw	LL	p_{varsel} = (100, 200, 500, 1,000,	0.055	0.999	0.273	0.672
tRF_{BS}+LR		RF	+30 s (LR)	1 [2]	randomForest	Platt LR	BS	, 2,000, 5,000, 7,500, 10,000)	0.056	0.997	0.086	0.266
tRF_{ME}+LR	9	RF	+30 s (LR)	1 [2]	randomForest	Platt LR	ME	nodesize = 1	0.042	0.998	0.062	0.156
tRF_{LL} +LR		RF	+30 s (LR)	1 [2]	randomForest	Platt LR	LL	nodesize = 1	0.058	0.995	0.089	0.291
tRF_{BS} +FLR		RF	+8-9 min (FLR)	1 [2]	randomForest	Platt Firth	BS	nodesize = 1	0.054	0.997	0.086	0.194
tRF_{ME} +FLR	8	RF	+8-9 min (FLR)	1 [2]	randomForest	Platt Firth	ME	nodesize = 1	0.037	0.999	0.062	0.150
tRF_{LL} +FLR		RF	+8-9 min (FLR)	1 [2]	randomForest	Platt Firth	LL	nodesize = 1	0.056	0.999	0.089	0.205
tRF_{BS} +MR		RF	+7-8 min (MR)	11 [2]	randomForest	MR	BS	nodesize = 1	0.051	0.997	0.082	0.176
tRF_{ME} +MR	4	RF	+7-8 min (MR)	11 [2]	randomForest	MR	ME	nodesize = 1	0.027	0.999	0.046	0.095
tRF_{LL} +MR		RF	+7-8 min (MR)	11 [2]	randomForest	MR	LL	nodesize = 1	0.055	0.999	0.086	0.188

- **ELNET**
    - It used 1,000 most variable CpG probes
    - ME ranked 8th, AUC ranked 5th
    - ME (2.7%), BS (0.048) and LL (0.109) and negligibly low AUC (99.9 %)

ELNET (1k)	7	ELNET	~7.5 h (12-15 min/fold)	31 [4]	glmnet	raw	ME	$\alpha = 0 \mid 0.025$; $\lambda =$ (0.0010-0.0036)	0.032	0.999	0.059	0.131
ELNET (10k)	5	ELNET	~72 h (2-2.25 h/fold)	31 [4]	glmnet	raw	ME	$\alpha = 0$; $\lambda =$ (0.012-0.038)	0.027	0.999	0.048	0.109

- **SVM**
    - More effective ME = 2.1% (lowest overall) with Platt scaling with Firth regression
    - While simple LR can be more effective to improve BS (second) and LL (fourth) by 8-9 times respectively
    - MR (`SVM-LK+MR`) achieves the most comprehensive improvement across all metrics. It reduced BS by a factor of 9.5 and LL by a factor of 11.5, resulting in the second lowest ME (2.1%) and AUC (99.9%), lowest BS (0.039) and lowest LL (0.085)

SVM-LK		SVM	~28 h (50-70 min/fold)	11 [3]	e1071	raw	ME	C = 0.001 | 0.01	0.032	0.999	0.372	0.978
SVM-LK+LR	2	SVM	+30 s (LR)	1 [2]	e1071	Platt LR	ME	C = 0.001 | 0.01	0.025	0.999	0.043	0.112
SVM-LK+FLR	3	SVM	+8-9 min (FLR)	1 [2]	e1071	Platt Firth	ME	C = 0.001 | 0.01	0.021	0.999	0.044	0.135
SVM-LK+MR	1	SVM	+7-8 min (MR)	11 [2]	e1071	MR	ME	C = 0.001 | 0.01	0.021	0.999	0.039	0.085
SVM-LK (GPU)	6	SVM	~5 h	1080Ti	Rgtsvm-GPU	global softmax	ME	C = 0.01 | 0.001 ; n.SV= 1,300-1,600	0.033	0.998	0.056	0.144
SVM-CS[6]		SVM	~6 h (13-15 min/fold)	7 [1]	LiblineaR	-	ME	C ≥ 0.001	0.028	-	-	-

- **Boost Tree**
    - Boosted model using ME as evaluation metric outperforms model using LL
    - Overall ME of 5.1% and AUC of 99.9%, with the second lowest BS (0.15) and LL (0.43) among the base ML classifiers studied

XGBoost		BT	~65-70 h (110-130 min/fold)	72 [5]	xgboost	raw	ME	Tables 3 and 4	0.051	0.999	0.150	0.430
XGBoost+LR		BT	+30 s (LR)	1 [2]	xgboost	Platt LR	ME	Tables 3 and 4	0.055	0.991	0.087	0.452
XGBoost+FLR		BT	+8-9 min (FLR)	1 [2]	xgboost	Platt Firth	ME	Tables 3 and 4	0.053	0.993	0.089	0.384
XGBoost+MR		BT	+7-8 min (MR)	11 [2]	xgboost	MR	ME	Tables 3 and 4	0.046	0.999	0.092	0.247

Other Issue