ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

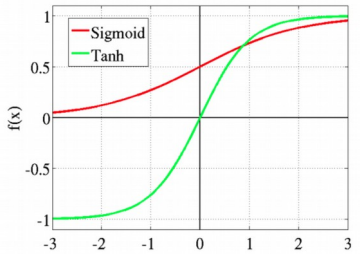
CVL Computer Vision Lab

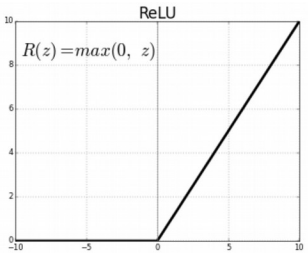
Adaptive Activation Network and Functional Regularization for Efficient and Flexible Deep Multi-Task Learning (AAAI-2020)

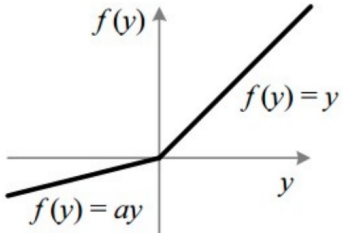
Reading Group Dec. 11, 2019

Suman Saha (postdoc), **Computer Vision Lab @ ETH Zurich**

Motivation: DNN Activation Functions





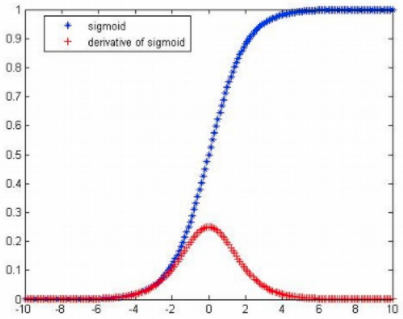


- Learning activation functions to improve deep neural networks (DNNs) [1]
- Parameters in the linear components (W and b) are learned from data
- While nonlinearities are predefined, e.g. sigmoid, tanh or ReLU etc.
- Assumption – an arbitrary complex function can be approximated using any of these common nonlinear functions
- In practice, the choice of nonlinearity affects:
  - ⟶ the learning dynamics
  - ⟶ network expressive power

[1] Agostinelli, Forest, et al. "Learning activation functions to improve deep neural networks." arXiv preprint arXiv:1412.6830 (2014).

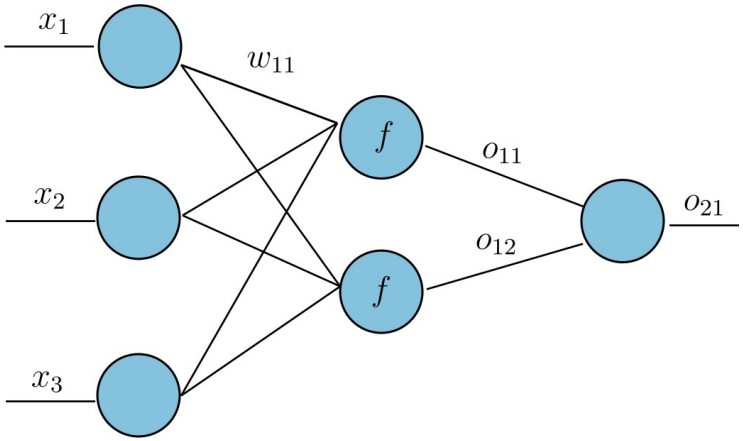
Motivation: Choice of Nonlinearity

- Active research area – design activation functions that enable fast training of DNN



Vanishing Gradient Problem

- Derivative of a Sigmoid Function
- ranges between **0 to 0.25**



Weight Update

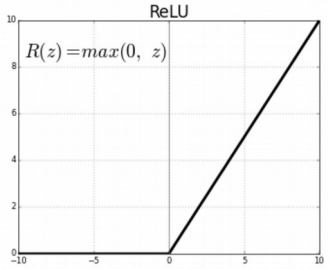
$$w_{11}^{new} = w_{11}^{Old} - \eta \frac{\partial L}{\partial w_{11}^{Old}}$$

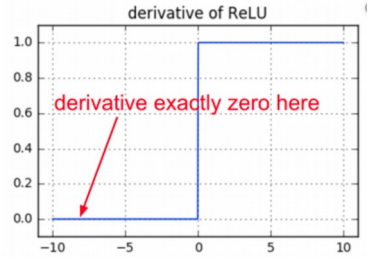
$$\frac{\partial L}{\partial w_{11}} = \frac{\partial o_{21}}{\partial o_{11}} \cdot \frac{\partial o_{11}}{\partial w_{11}}$$

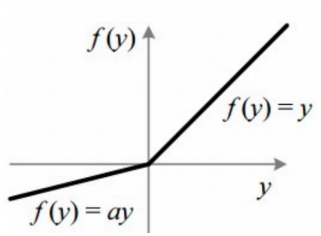
- For DNN with more layers, the gradients tend to vanish more in the lower layers

Motivation: Choice of Nonlinearity

- The rectified linear activation function (ReLU) does not saturate like sigmoidal functions
- helps to overcome the vanishing gradient problem







Anohter recent activation functions

- Maxout activation (Goodfellow et al., 2013) – computes the maximum
- of a set of linear functions
- Springenberg & Riedmiller (2013) replaced the max function
- Gulcehre et al. (2014) explored an activation function that replaces the max function with an L_p norm

Motivation

- The type of activation function can have a significant impact on learning
- One way to explore the space of possible functions is to learn the activation function during training (Agostinelli et al., 2014)

Adaptive Piecewise Linear (APL) units

- Activation functions as a sum of hinge-shaped functions resulting a piecewise linear activation function

$$h_i(x) = \max(0, x) + \sum_{s=1}^{S} a_i^s \max(0, -x + b_i^s)$$

- S (the **number of hinges**) is a hyperparameter set in advance
- a_i^s, b_i^s are the **learnable parameters**, where $i \in 1, ..., S$
- a_i^s variables control the **slopes** of the linear segments
- b_i^s variables determine the **locations** of the hinges

Adaptive Piecewise Linear (APL) units

$$h_i(x) = \max(0, x) + \sum_{s=1}^{S} a_i^s \max(0, -x + b_i^s)$$

- Fig.1 shows example APL functions for S = 1
- for large enough S, APL can approximate arbitrarily complex continuous functions
- the first term in Eq. (1) is **ReLU**
- when **x < 0** the **derivative** of ReLU is **0** resulting **dead neurons**

Leaky ReLUs addresses the dead neurons problems, e.g. leaky ReLU may have y = 0.01x when x < 0

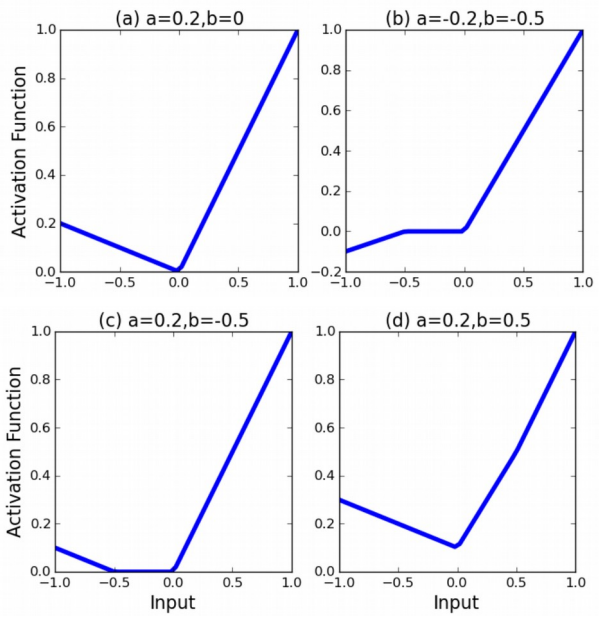


Figure 1: Sample activation functions obtained from changing the parameters. Notice that figure b shows that the activation function can also be non-convex.

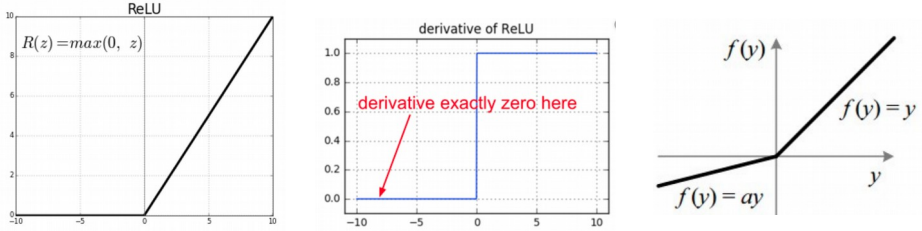
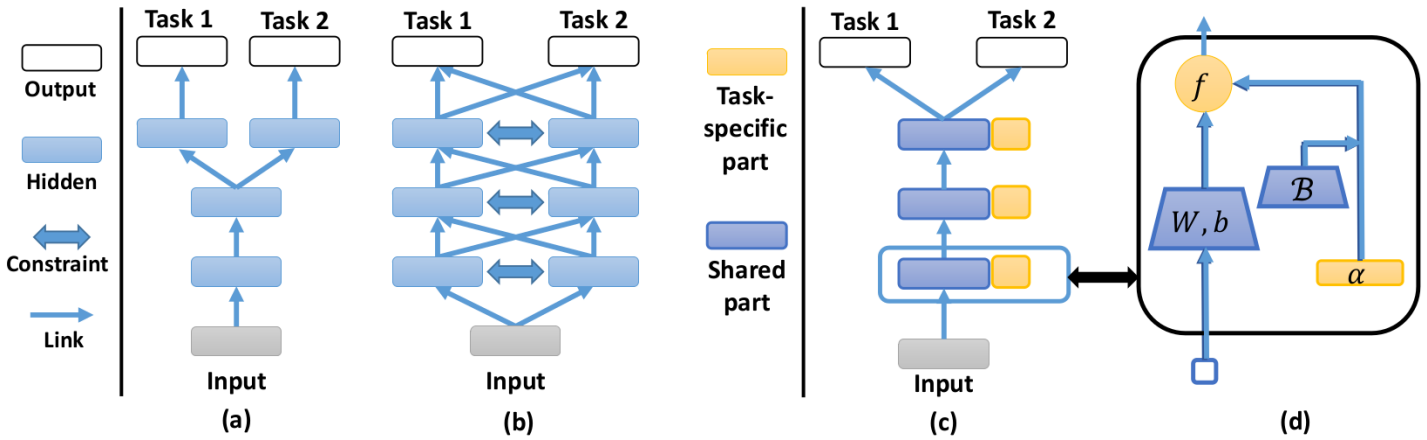


Figure 2

TAAN (Task Adaptive Activation Network)



Categories of deep learning MTL: (a) Hard-sharing; (b) Soft-sharing; (c) Task Adaptive Activation Network (proposed model); (d) Inner Structure of Adaptive Activation Layer.

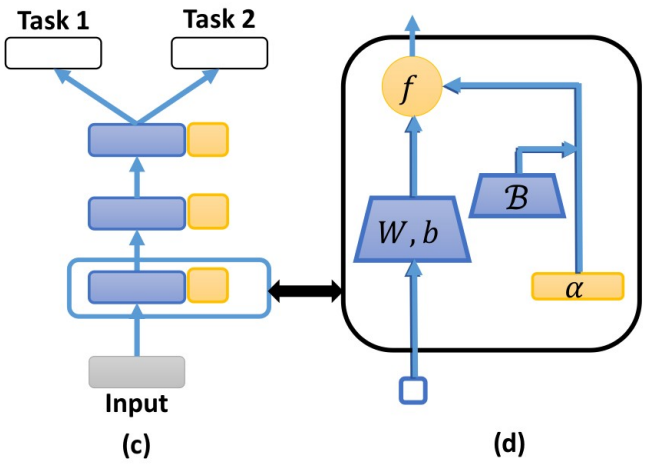
- **Proposed approach = hard-sharing + learnable task-specific activation functions**
- all tasks can share their weights and biases on the hidden layers
- more **scalable** than the soft-sharing methods where the number of network components is proportional to the number of tasks

TAAN

- For a task t, given the input from either the previous layer or data input, the output of the l-th AAL (Adaptive Activation Layer) is defined by

$$h_l^t = \mathcal{F}_l^t(W_l h_{l-1} + b_l),$$

- weight and bias parameters are shared across tasks



(c) Task Adaptive Activation Network (proposed model);
(d) Inner Structure of Adaptive Activation Layer.

- The task-specific activation function for ***task t*** and ***layer l*** is defined as

$$\mathcal{F}_l^t(\cdot) = \sum_{i=1}^{M} \alpha_l^t(i)\mathcal{B}_i(\cdot),$$

- Recall from slide 6 and 7, M (the number of hinges) is a hyperparameter set in advance
- $\alpha_l^t = [\alpha_l^t(1), \cdots, \alpha_l^t(M)] \in \mathbb{R}^M$ denotes the coordinates of the basis functions $\{\mathcal{B}_i\}_{i=1}^M$

TAAN

- Each task has its own coordinate vector $\alpha_l^t \in \mathbb{R}^M$
- there is a **coordinate matrix** $\alpha_l \in \mathbb{R}^{T \times M}$ attached to each AAL hidden layer
- the coordinate matrices of the hidden layers **control the level of network sharing** among multiple tasks
- For instance, if tasks 1 and 2 have **more shared knowledge** at the 1st hidden layer, $\mathcal{F}_1^1(\cdot)$ and $\mathcal{F}_1^2(\cdot)$ have **higher similarity**,
- thus α_1^1 and α_1^2 are more similar
- On the other hand, if tasks 1 and 2 share **less knowledge** at the 2nd hidden layer, **their activation functions** are **more diverse**
- During the training phase, the coordinate matrices of all hidden layers are optimized to extract both the shared and task-specific knowledge from data

Metrics for Activation Functions

- In order to understand how TAAN captures the relationship of multiple tasks, we need a metrics to measure the **difference/similarity** between **two activation functions**
- As the basis functions of APL units are unbounded and non-orthonormal, the coordinate vectors do not reveal much property about the functions
- Besides, the commonly used L^2 norm and inner product are infinite almost everywhere, it is impossible to use them as metrics
- Authors redefine the finite inner product and norm assuming X is a random variable with Gaussian distribution $p(x) = \mathcal{N}(\mu, \sigma^2)$

Functional Regularization

- For each layer of TAAN, the coordinate matrix $\alpha_l \in \mathbb{R}^{T \times M}$ can be learned directly from the training data
- As the tasks in MTL are generally considered to be related, it is reasonable to encourage sharing more than splitting
- This insight is incorporated into TAAN by introducing regularization term on α_l during training
- Authors propose two functional regularization methods to further enhance the performances of TAAN

Functional Regularization

Baseline: Trace-Norm

- The first regularization hypothesis is that the matrix $\alpha_l \in \mathbb{R}^{T \times M}$ is low-rank, as the tasks in MTL often have high correlation
- Thus, authors introduce a regularization term to α_l

$$\mathcal{L}_{tn}(\alpha_l) = trace(\sqrt{\alpha_l \alpha_l^T})$$

- $\sqrt{\cdot}$ denotes the square root of matrix

Functional regularization by cosine similarity

the similarity of two task-specific activation functions can be defined by the cosine similarity, which is computed as:

$$\mathcal{L}_{cos}(\alpha_l) = -\frac{1}{T^2}\sum_{ij}\mathcal{C}_{ij}(\alpha_l),$$

$$\mathcal{C}_{ij}(\alpha_l) = \frac{\langle \mathcal{F}_l^i, \mathcal{F}_l^j \rangle}{\sqrt{\langle \mathcal{F}_l^i, \mathcal{F}_l^i \rangle \langle \mathcal{F}_l^j, \mathcal{F}_l^j \rangle}},$$

Functional Regularization

Functional regularization by distance

- Given the coordinate matrix for the l-th layer of network, authors compress the distance function Eq. (3) between task-specific activation functions with the following regularization

$$\mathcal{L}_{dis}(\alpha_l) = \frac{1}{T^2}\sum_{ij}\mathcal{D}_{ij}(\alpha_l),\ \ \mathcal{D}_{ij}(\alpha_l) = d^2(\mathcal{F}_l^i, \mathcal{F}_l^j).$$

- the training loss of a TAAN with L task-specific activation layers becomes

$$\mathcal{L}_{\text{total}} = \sum_{t=1}^{T}\mathcal{L}_t\Big(\mathcal{M}_t, \{x_i^t, y_i^t\}\Big) + c\sum_{l=1}^{L}\mathcal{L}_{\text{MTL}}(\alpha_l),$$

- Where $\mathcal{L}_{\text{MTL}} \in \{\mathcal{L}_{tn}, \mathcal{L}_{dis}, \mathcal{L}_{cos}\}$ and c is the regularization coefficient

Functional Regularization

Functional regularization by distance

- Given the coordinate matrix for the l-th layer of network, authors compress the distance function Eq. (3) between task-specific activation functions with the following regularization

$$\mathcal{L}_{dis}(\alpha_l) = \frac{1}{T^2}\sum_{ij}\mathcal{D}_{ij}(\alpha_l),\ \ \mathcal{D}_{ij}(\alpha_l) = d^2(\mathcal{F}_l^i, \mathcal{F}_l^j).$$

- the training loss of a TAAN with L task-specific activation layers becomes

$$\mathcal{L}_{\text{total}} = \sum_{t=1}^{T}\mathcal{L}_t\Big(\mathcal{M}_t, \{x_i^t, y_i^t\}\Big) + c\sum_{l=1}^{L}\mathcal{L}_{\text{MTL}}(\alpha_l),$$

- Where $\mathcal{L}_{\text{MTL}} \in \{\mathcal{L}_{tn}, \mathcal{L}_{dis}, \mathcal{L}_{cos}\}$ and c is the regularization coefficient

Experiments

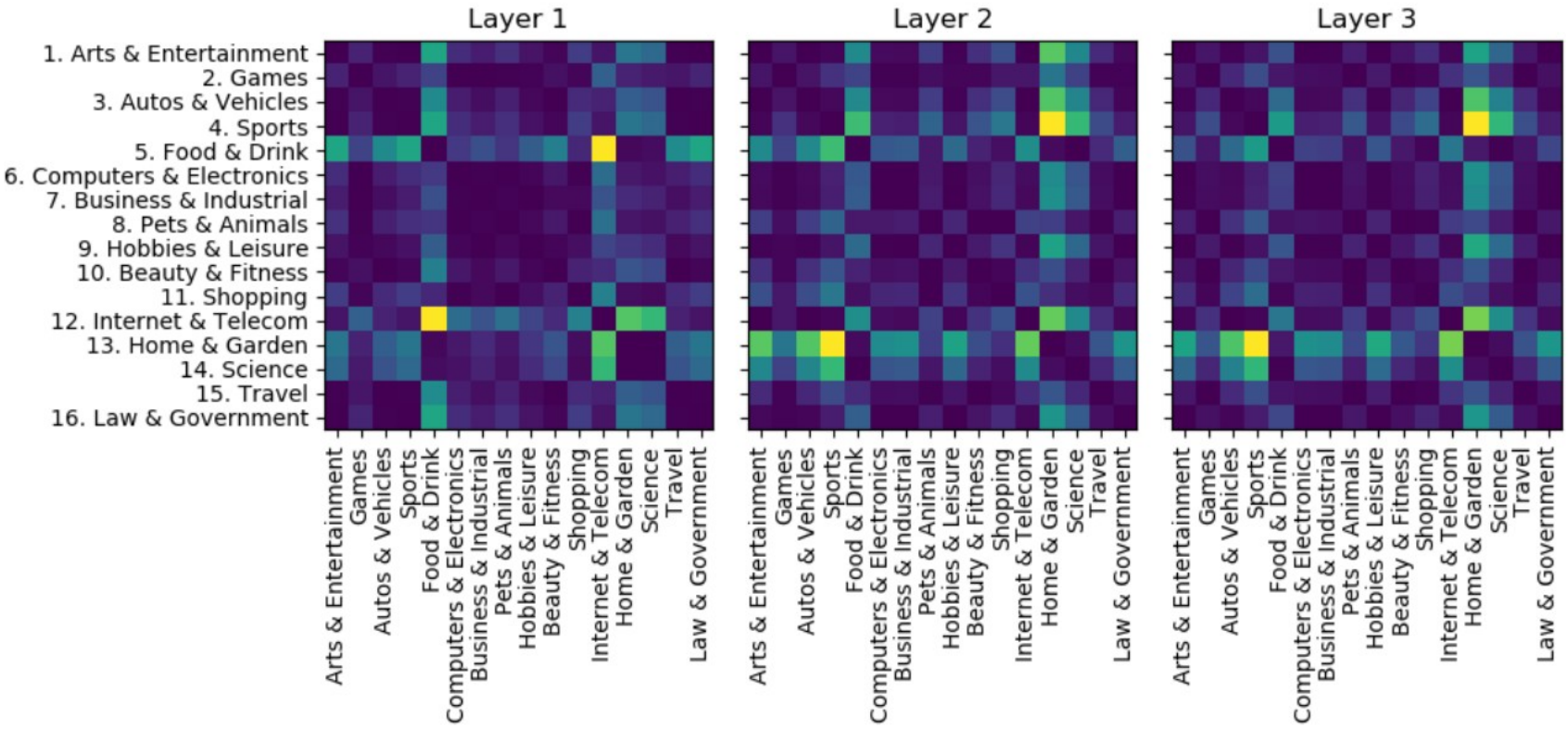
Multi-Domain Multi-Label Classification

- Conduct experiments on Youtube-8M, a large dataset that consists of over 6.1 billion of Youtube videos. Each video has multiple labels from a vocabulary of 3800 topical entities, which can be further grouped into 24 top-level categories.
- To create an MTL experiment, authors consider each top-level category as a specific domain.
- For each domain, they have to define a multi-label classifier to recognize various attributes of the data

Experiments

Multi-Domain Multi-Label Classification

- The task IDs and their corresponding domains



Experiments

Multi-Domain Multi-Label Classification

Table 1: The number of network parameters and classification performance on the Youtube-8M dataset

	# of params	Task mAP%10																Mean mAP%10
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
Benchmarks																		
STL	56.1 M	.683	.807	.835	.689	.781	.431	.559	.831	.596	.779	.551	.916	.505	.383	.699	.759	0.675
Hard-Share	**6.93** M	.770	.838	.870	.806	.824	.598	.676	.846	.754	.808	.716	.940	.629	.680	.806	.839	0.775
Soft-Order	**6.93** M	.785	.845	.877	.819	.825	.611	.683	.848	.764	.809	.727	.943	.648	.694	.819	.838	0.783
MRN_t	56.1 M	.664	.806	.829	.608	.778	.413	.502	.828	.576	.775	.542	.916	.383	.355	.687	.695	0.647
MRN_{full}	61.6 M	.670	.804	.828	.607	.777	.416	.506	.828	.578	.775	.543	.916	.383	.355	.688	.696	0.648
DMTRL-Tucker	39.1 M	.660	.817	.812	.651	.794	.473	.490	.793	.569	.742	.594	.944	.475	.475	.697	.710	0.669
DMTRL-LAF	20.1 M	.851	.869	.890	.875	.841	.691	.757	.873	.839	.854	.809	.948	.746	.792	.875	.896	0.838
DMTRL-TT	46.2 M	.860	.874	.893	.888	.846	.710	.778	.879	.849	**.860**	.822	**.950**	**.770**	**.807**	**.881**	**.903**	0.848
TAAN (Ours)																		
TAAN	**6.93** M	**.882**	**.896**	**.910**	**.902**	**.857**	**.739**	**.804**	.879	**.859**	.828	.808	.934	.717	.774	.833	.848	0.842
TAAN + $\mathcal{L}_{tn}$	**6.93** M	.741	.824	.861	.779	.813	.572	.636	.831	.711	.785	.681	.937	.581	.637	.783	.809	0.749
TAAN + $\mathcal{L}_{cos}$	**6.93** M	**.896**	**.906**	**.915**	**.915**	**.859**	**.769**	**.830**	**.885**	**.879**	.843	**.828**	.937	.756	.805	.854	.876	**0.860**
TAAN + $\mathcal{L}_{dis}$	**6.93** M	**.889**	**.899**	**.912**	**.910**	**.859**	**.752**	**.820**	**.886**	**.873**	.836	.821	.933	.731	.789	.845	.863	**0.851**

Experiments

Multi-Domain Multi-Label Classification

Table 2: Training and inference speed evaluation on Youtube-8M (Note: the dimension of hidden layer is 512)

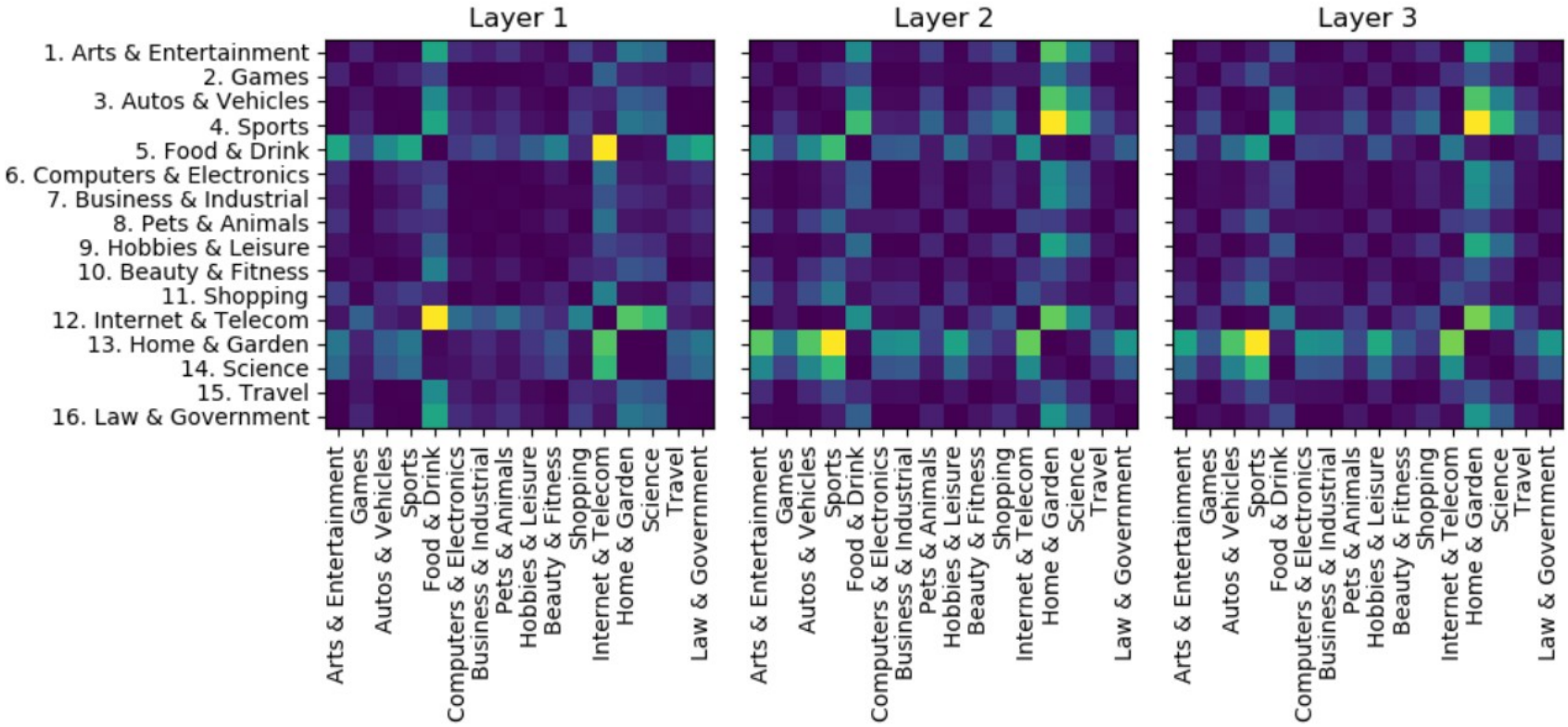
Model	training time↓	inference time↓
MRN_t	156.1 s	8.83 s
MRN_{full}	157.6 s	8.83 s
DMTRL-Tucker	66.02 s	1.53 s
DMTRL-LAF	24.96 s	0.30 s
DMTRL-TT	**52.75 s**	**1.44 s**
TAAN	20.05 s	0.88 s
TAAN + $\mathcal{L}_{tn}$	22.86 s	0.88 s
TAAN + $\mathcal{L}_{cos}$	**40.34 s**	**0.88 s**
TAAN + $\mathcal{L}_{dis}$	38.35 s	0.88 s

Experiments

Visualization

- TAAN is able to capture the complicated knowledge sharing for the tasks on the Youtube-8M dataset. For instance, domain “Food & Drink” shares all the hidden layers with domain “Home & Garden”. TAAN also discover that the domains “Food & Drink” and “Internet & Telecom” are the most unrelated, as the distances between their activation functions are always high.

Distance matrices of the activation functions in TAAN. Light colors denote less similarity.



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

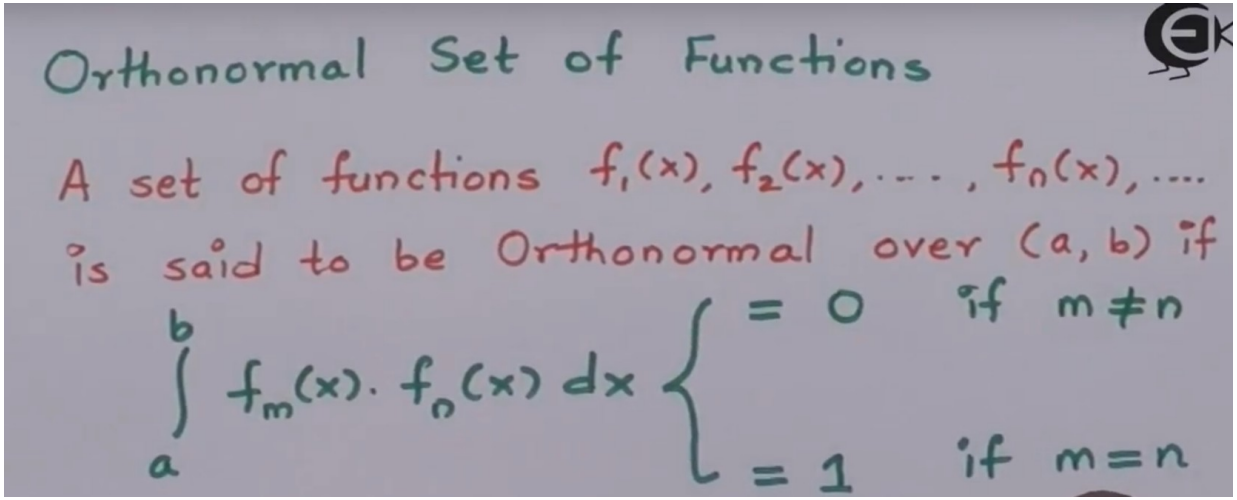
Questions?

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CVL Computer Vision Lab

Thank you for your attention!

References

- 
Orthonormal Set of Functions
A set of functions $f_1(x), f_2(x), \ldots, f_n(x), \ldots$
is said to be Orthonormal over (a, b) if
$\int_a^b f_m(x) \cdot f_n(x)\, dx$
$= 0$ if $m \neq n$
$= 1$ if $m = n$