Biased Mixtures Of Experts: Enabling Computer Vision Inference Under Data Transfer Limitations

Alhabib Abbas and Yiannis Andreopoulos

***Abstract*—We propose a novel mixture-of-experts class to optimize computer vision models in accordance with data transfer limitations at test time. Our approach postulates that the minimum acceptable amount of data allowing for highly-accurate results can vary for different input space partitions. Therefore, we consider mixtures where experts require different amounts of data, and train a sparse gating function to divide the input space for each expert. By appropriate hyperparameter selection, our approach is able to bias mixtures of experts towards selecting specific experts over others. In this way, we show that the data transfer optimization between visual sensing and processing can be solved as a convex optimization problem. To demonstrate the relation between data availability and performance, we evaluate biased mixtures on a range of mainstream computer vision problems, namely: (i) single shot detection, (ii) image super resolution, and (iii) realtime video action classification. For all cases, and when experts constitute modified baselines to meet different limits on allowed data utility, biased mixtures significantly outperform previous work optimized to meet the same constraints on available data.**

***Index Terms*—mixtures of experts, constrained data transfer, single shot object detection, single image super resolution, realtime action classification.**

I. Introduction

When enough data is provided at test time, deep neural networks perform well for a wide range of challenging computer vision tasks. This is true especially for large models, as it is now well understood that the performance of neural networks scales with the number of trainable weights and the dimensionality of inputs processed during inference [19], [20]. However, the precondition of data availability at test time is only possible when visual sensors and learned inference models coexist in hardware, which excludes cases where data is collected from sensors to be transferred and processed in remote environments (e.g., by powerful servers located within data-centers). To bridge the gap between the input requirements of models that exist in such contexts, it is important to design models that can perform well when available communication resources are limited between the visual sensing and neural network processing parts of the system. For instance, cloud-based visual analysis, remote medical imaging, low-latency game streaming services, and drone or Internet-of-Things oriented computer vision [9] [29], [46], [55], have stringent constraints on the amount of data that can be provided between data-producing clients and data-consuming models on cloud servers. In order to bring computer vision models to wider practical use, it is therefore imperative to provide a solution to data availability constraints at test time.

The authors are with the Electronic and Electrical Engineering Department, University College London, London, UK, University College London, Roberts Building, WC1E 7JE, UK (e-mail: {alhabib.abbas.13, i.andreopoulos@ucl.ac.uk).

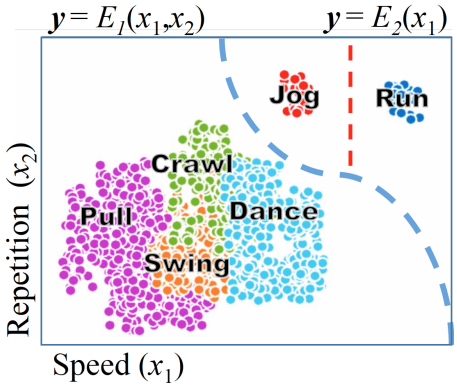


Fig. 1: Sample space of a classification task using two features (speed and repetition of motion), where colours indicate different actions. The blue line shows an instance of a learnable input space partition, and the red line shows a one-dimensional classification boundary learnable by a designated expert $E_2(x_1)$ with reduced data requirement.

Since deep learning models typically require a fixed amount of data for inference regardless of the specific nature of inputs to process, this leads to unnecessary and often unachievable demands in the amount of required data traffic for remote inference. Although some work has been devoted to input dimensionality reduction [18], [28], [50] and rate-constrained model optimization for specific tasks [21], [55], to the best of our knowledge, no task-agnostic method has been proposed that explicitly addresses data scarcity at test time by considering the variance between different domains in input space. The example of Figure 1 illustrates a classification task where the acceptable data cost of inference can vary for different input space partitions. That is, two features (speed and repetition of motion) can be used to classify the bottom-left examples in Figure 1, while one feature suffices for distinguishing "Jog" examples from "Run" examples on the top-right. Reducing the retained dimensions directly correlates with the *data cost* of inference. To leverage inherent variances across different input space partitions, and by selecting among two experts E_1 and E_2 which respectively require d_1 and d_2 bytes per input where $d_1 > d_2$, decision boundaries can be determined to appropriately pass more data for more difficult inputs. Learning decision boundaries similar to those of Figure 1 can allow sensors to remotely communicate data as necessary, subject to the general position of an input within its respective space. This reduces the overall data cost for inference that

is accurate enough for the task at hand. Consequentially, this can relieve unnecessary load on communication resources that exist between sensors and remote machines used for visual inference. Our work proposes a solution to learning such decision boundaries directly from data for any model wherein inputs can be subsampled or reduced, and for any specified limit on data cost. Our contributions are summarised below:

1) We introduce a novel class of mixtures-of-experts, wherein some experts are favored to others by design. When experts of different data requirements are included, this allows mixtures to meet different constraints on allowed data utility.
2) We propose two methods to train biased mixtures such that input space is effectively partitioned for each expert to realize data-efficient mixtures.
3) We show that data transfer optimization between visual sensing and processing can be formulated as a convex optimization problem, and present an ablation study of the benefit of biased mixtures under different contexts of allowed limits on data utility.

The expert utility biasing method proposed in this paper can be applied to reduce the data cost of any model wherein the size of inputs can be subsampled or reduced. To illustrate this, we train and validate on a variety of tasks spanning multiple domains. Specifically, we validate on the tasks of: single shot object detection from the work of Wei *et. al* [25], realtime video action classification from the work of Zhang *et. al* in [53] and Jubran *et. al* [8], and image super resolution from the work of Shi *et. al* [42] and Dong *et. al* [11]. The remainder of this paper is organized as follows: In Section II, we give an overview of recent work on rate and complexity optimization. Section III details the proposed biased expert selection and describes its general architecture and how it is trained. In Section IV we evaluate the performance of the proposed method on all tasks, and illustrate the benefits that biased mixtures of experts can provide on multiple models for each task. Finally, Section V summarises our findings and outlines possible directions for future work.

II. Related Work

Within the field of compact image representation, and in order to communicate data-efficient codes across networks for remote processing, directly engineered compression techniques were extensively studied to culminate in existing image compression standards [32], [36]. More recently, learned methods [33], [49], [50] have attracted attention as the next step towards more data-driven image compression. Salient among recent advances in this domain are variational autoencoders [2], [31], [37] and adversarial models [10], [14], [35]. In order to adapt learned codes to arithmetic coders, state-of-the-art proposals on learned compression [30], [34], [38], [49] additionally learn context models to predict posteriors of latent code components conditional on all preceding components. Specifically, and to move learned compression closer to replacing established coders [32], [36], context models [30], [38] use tractable masked convolutions to regulate entropies of obtained image representations such that they can be coded more effectively by subsequent entropy coders. In distributed systems of visual analysis, and in order to reduce throughput requirements on input, latent states of learned image reconstruction machines [2], [14], [31], [35] and entropy regulated compressors [30], [33], [38], [49] can be used instead of full-length inputs as representative signals to remote inference models.

Other studies consider the regulation of input volumes for *complexity* optimization, and propose modifications that are applicable to a wide range of models. In this realm, proposals such as static model pruning [15], [16], [19], reduce complexity by modifying models in a persistent manner for all inputs at test time. More recent proposals [3], [4], [23], [41] show how the test-time complexity of very large networks can be substantially reduced by conditioning computation to the content of feature maps at runtime, and do so by training external agents to enable or disable different parts of models subject to the unique properties of each input. However, all of the aforementioned works optimize solely for complexity, and always consider the maximum amount of input to be available at test time. Other proposals also studied specific vision tasks in order to reduce the data requirement of deep neural network models. For example, this can be seen in previous work [8], [53], [55], where input volumes are reduced by distilling input sequences to their most useful elements before relaying to remote servers for semantic analysis. Other work [22], [52] mainly focused on task-specific mappings of inputs onto lower-dimensional space before training with more data-efficient models, and recent advances in domain adaptation and transfer learning [26], [39], [48] can also be used to learn compressed codes tuned to particular models. However, for any specified source distribution, domain adaptation [26], [39], [48] and other proposals mentioned above [8], [53], [55] equally compact all sampled inputs to fixed length codes, and varying degrees of entropy among input examples are ignored. As such, low-entropy inputs (which contain less information relative to others) are mapped to redundantly long code-lengths, and subsequently incur unnecessary loads on data transfer assets and inference complexity. In this sense, while the aforementioned advances are important in determining useful transformations to adaptive or fixed-length codes, complementary techniques are necessary to determine required code lengths prior to compression and inference.

In our work, we consider the *data cost* optimization problem in task-agnostic manner, and determine required input volumes *prior to visual inference*. Specifically, we consider how input space partitions vary in the amount of data required per input in order to ensure good performance, and leverage this variance to train more data-efficient mixtures of experts. To do so, we take inspiration from recent work [19], [23], [41] to propose a mixture of experts where expert utility is biased towards specific experts. While meeting predefined constraints on expert utility bias, we train a sparse gating function to select the most adequate expert to use from a set of experts of varied input requirements. Importantly, our method does not modify any pre-existing methods for complexity optimization or task-specific data cost reduction. As such, our proposal can be applied in conjunction with recent proposals on learnable compression [30], [34], [38] and domain adaptation [26], [39],

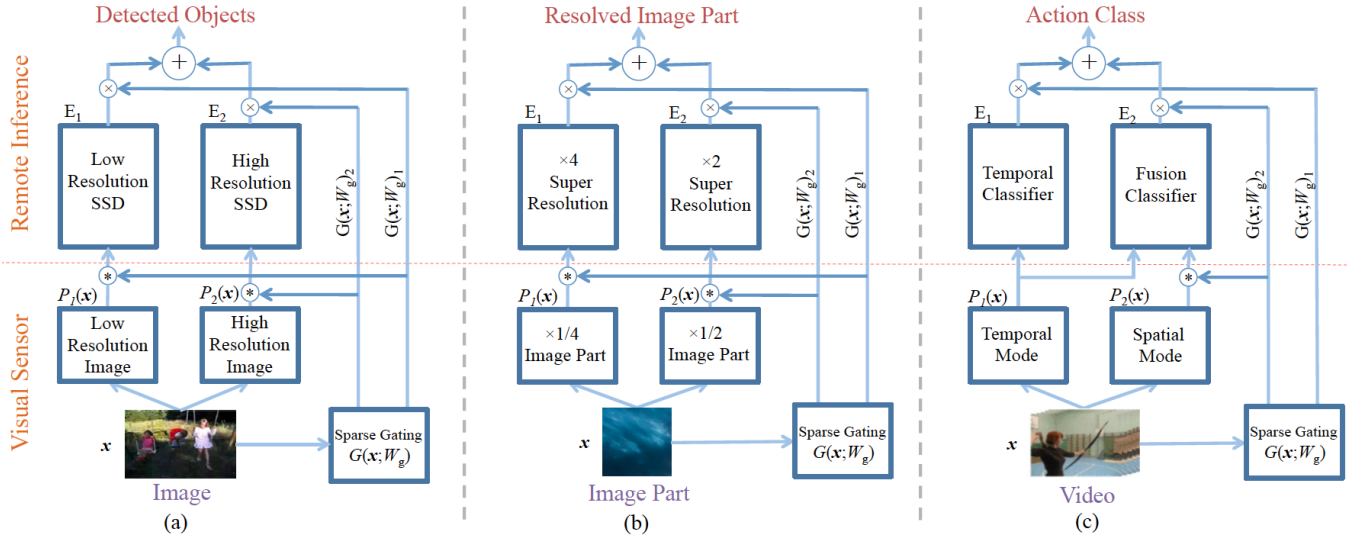


Fig. 2: An illustration of how biased mixtures of experts can be applied for different computer vision tasks. $(*)$ is a special operator that transmits data to remote inference parts of the model whenever it receives a non-zero gate value. From left to right: (a) single shot detection (SSD), (b) image super resolution, and (c) realtime action classification.

[48] to reduce the data cost of visual inference. We show that our method can be augmented in accordance with any set of pre-trained experts to partition input space such that constraints on data availability are met at test time, while providing the best possible accuracy of inference.

III. Biased Expert Selection

A. General Architecture Formulation

Let $\mathcal{E}$ denote a mixture of N experts where $\mathcal{E} = \{E_1, E_2, ..., E_N\}$, and each expert E_n is a modified variant of a task-performing baseline model. Per input $\boldsymbol{x}$, a gating function determines the contribution of the n^{th} expert as:

$$G(\boldsymbol{x}; \mathcal{W}_g)_n = \frac{e^{f(\boldsymbol{x};\mathcal{W}_g)_n}}{\sum_{m \neq n}^{N} e^{f(\boldsymbol{x};\mathcal{W}_g)_m}} \tag{1}$$

where $\mathcal{W}_g$ is a set of trainable weight parameters, $m \in \mathbb{N}$ denotes remaining gate indices, and $f(\boldsymbol{x}; \mathcal{W}_g) \in \mathbb{R}^N$ is the output of a specified gating model (e.g, a multi-layer perceptron). The output $\boldsymbol{y}$ of the mixture of experts is:

$$\boldsymbol{y} = \sum_{n=1}^{N} G(\boldsymbol{x}; \mathcal{W}_g)_n E_n(P_n(\boldsymbol{x})) \tag{2}$$

where P_n is a preprocessing function to accommodate $\boldsymbol{x}$ for the n^{th} expert (e.g., P_n performs subsampling if E_n ingests sub-sampled inputs). Mixtures of experts are typically trained using a task loss that calculates the error between a provisioned ground-truth and $\boldsymbol{y}$. In our proposed Biased Mixtures-of-Experts (BMoE), experts are activated only when needed, and activating some experts is more favorable to activating others. In addition, all experts are optimized before training the mixture, and the training loss is back-propagated through the gating function exclusively during training. In Figure 2 we illustrate some examples of how biased mixtures can be applied for different tasks.

To adjust mixtures for biased expert selection, we denote the desired amount of bias in expert selection by $\boldsymbol{b}$, where each of its components b_n specifies per batch the ratio of input examples to pass to each n^{th} expert. Importantly, elements of $\boldsymbol{b}$ denote *frequencies* of use as ratios and cannot be assigned negative values (e.g., setting $b_n = 0.1$ to use expert E_n 10% of the time), giving the properties $0 \leq b_n \leq 1$, and $||\boldsymbol{b}||_1 = 1$. We consider two methods of training for biased expert selection: *(i)* a soft regularization approach where a regularization term is included in the total loss to encourage bias, and *(ii)* fixing the average data cost *per batch*, by enforcing a constant number of training examples to each expert in accordance with $\boldsymbol{b}$ and training only with respect to the task loss. Both methods encourage mixtures of experts to maximize performance while meeting the specified bias, and we describe in detail each method in the following:

B. Soft Bias Regularization

When using soft bias regularization, the most suitable expert to use is selected *per input* via a sparse gating function, and all other experts are omitted. To do so, akin to [41] for each input $\boldsymbol{x}$ only the expert associated with the highest gate value is considered for inference, and we write the sparse gating function as:

$$G(\boldsymbol{I}; \mathcal{W}_g)_n = \psi(f(\boldsymbol{x}; \mathcal{W}_g))_n \cdot \frac{e^{f(\boldsymbol{x};\mathcal{W}_g)_n}}{\sum_{m \neq n}^{N} e^{f(\boldsymbol{x};\mathcal{W}_g)_m}} \tag{3}$$

where $\psi(f(\boldsymbol{x}; \mathcal{W}_g))$ is a non-linear operator which returns a one-hot vector indicating the top value in $f(\boldsymbol{I}; \mathcal{W}_g)$. From (3) we also define the utility of each n^{th} expert u_n as its total contribution per batch $\mathcal{X}$ comprising M examples:

$$u_n = \frac{1}{M} \sum_{\boldsymbol{x} \in \mathcal{X}} G(\boldsymbol{x}; \mathcal{W}_g)_n \tag{4}$$

and we calculate the bias regularization loss l_{bias} as a function of $\boldsymbol{u} \in \mathbb{R}^N$ and the specified bias vector $\boldsymbol{b}$:

$$l_{\text{bias}} = -w_{\text{bias}}\, log(1 - \frac{1}{\sqrt{2}}\, ||\boldsymbol{u} - \boldsymbol{b}||_2) \tag{5}$$

where w_{bias} is a hyperparameter to control the amount of bias to impose on the mixture. Since $\boldsymbol{u}$ and $\boldsymbol{b}$ describe frequencies as ratios and $||\boldsymbol{u}||_1 = ||\boldsymbol{b}||_1 = 1$, the distance $||\boldsymbol{u} - \boldsymbol{b}||_2$ is normalized by $\sqrt{2}$ to ensure the expression within the log function is always positive ($\sqrt{2}$ is the maximum possible euclidian distance between vectors with an L_1 norm of one). By applying the modifications to the gating function in (3), and including the bias regularization loss in (5) to the total loss, the mixture of experts is simultaneously trained to maximize task performance and meet the specified bias.

C. Batchwise Bias Enforcement

In our second proposal, rather than encourage mixtures to align the utility of their experts with the specified bias, we enforce bias *per batch* in accordance with $\boldsymbol{b}$, and train the mixture only with respect to its task loss. This in effect trains mixtures to make better expert selections for each input, while meeting the bias constraint for every batch. Specifically, with a batch size of M, batches are segmented such that Mb_n examples are passed to each n^{th} expert. To do so, starting from (1), we consider $G \in \mathbb{R}^{M \times N}$ as an M sized batch of gate vectors $G(\boldsymbol{x}; \mathcal{W}_g)$, and perform the procedure described in Algorithm 1. For each n^{th} expert, we denote gate values assigned to columns of input as $G_{:,n}$ and illustrate this in Figure 3.

Algorithm 1 Batchwise Bias Enforcement

Input: Soft gates batch $G \in \mathbb{R}^{M \times N}$

for $n = 1$ to $n = N$ **do**
  $K \leftarrow Mb_n$
  Calculate number of inputs to pass to the n^{th} expert
  $T \leftarrow TopK(G_{:,n}, K)$
  Find top K values corresponding to the n^{th} expert
  for $i = 1$ to $i = M$ **do**
    if $T_i \neq 0$ **then**
      $G_{i,j} \leftarrow 0 \;\; \forall j \neq n$
      For the i^{th} input, set all gate values not corresponding to n^{th} expert to 0
    else
      $G_{i,n} \leftarrow 0$
      Set gate value corresponding to the i^{th} input and n^{th} expert to 0
    end if
  end for
end for

D. Selecting Bias Values for Data Cost Optimization

So far, we discussed how biased mixtures are trained to make informed expert selections when a bias vector $\boldsymbol{b}$ specifies the frequency of expert utility. Here we detail our method for selecting useful biases that can optimize performance under different constraints on data utility. We consider the inference data cost vector $\boldsymbol{d} \in \mathbb{R}^N$, where each of its components d_n is the size of input volumes per example as seen by each expert (i.e., the data cost associated with $P_n(\boldsymbol{x})$). When mixtures are biased and an ample number of samples is considered, the average data cost is then expressed as $\bar{d} = \boldsymbol{b}\boldsymbol{d}^{\mathrm{T}} = \sum_{n=1}^{N} b_n d_n$. In this way, the biasing vector $\boldsymbol{b}$ can be tuned to allow for different average data costs of inference in the interval $[d_{min}, d_{max}]$, where d_{min} and d_{max} are the minimum and maximum amounts of data that can be ingested by experts in the mixture.

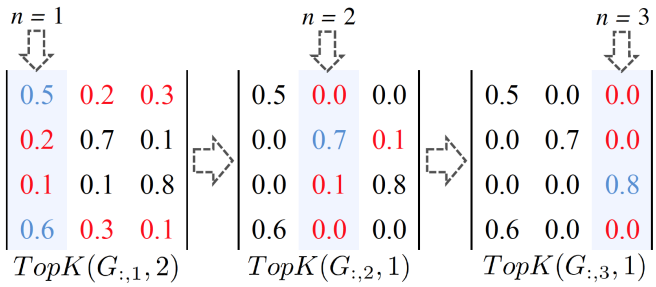


Fig. 3: Batchwise bias enforcement example when $N = 3$, $M = 4$ and $\boldsymbol{b} = [0.50, 0.25, 0.25]$. Inputs are selected per batch by iteratively sorting and selecting the top Mb_n highest gate values for each n^{th} expert. Gates subsequently set to zero are highlighted in red, and top (Mb_n) values are highlighted in blue.

Importantly, it can be seen that when $N > 2$ there can be multiple instantiations of $\boldsymbol{b}$ that produce the same average data cost $\bar{d}$. Thus, when an average data cost target $d_t \in [d_{min}, d_{max}]$ is specified, it is necessary to define a method by which to determine an appropriate bias vector $\boldsymbol{b}$ that is subsequently used in training biased mixtures. To address this, we consider $\boldsymbol{p} \in \mathbb{R}^N$ where p_n quantifies the performance of each optimized expert prior to inclusion in the mixture, and select $\boldsymbol{b}$ such that: *(i)* $\boldsymbol{b}$ satisfies $\bar{d} = d_t$, and *(ii)* $\boldsymbol{b}$ maximises the expected test performance as measured by $\boldsymbol{b}\boldsymbol{p}^{\mathrm{T}}$. That is, when each component p_n denotes an appropriate performance measure for the n^{th} expert on a designated set of inputs isolated from testing examples (e.g., p_n can be accuracy for classification tasks, or mean average precision for objection detection tasks), $\boldsymbol{b}\boldsymbol{p}^{\mathrm{T}}$ is a measure of performance when examples are randomly assigned to experts with respect to $\boldsymbol{b}$. In doing so, we reduce the problem of determining $\boldsymbol{b}$ for a specified data cost d_t to a linear optimization problem that achieves $\boldsymbol{b}\boldsymbol{d}^{\mathrm{T}} = d_t$, while maximising $\boldsymbol{b}\boldsymbol{p}^{\mathrm{T}}$. Since $||\boldsymbol{b}||_1 = 1$ and b_N can be expressed as $b_N = 1 - \sum_{n=1}^{N-1} b_n$. By expanding and substituting b_N, we get:

$$b_1 d_1 + b_2 d_2 + ... + (1 - \sum_{n=1}^{N-1} b_n) d_N = d_t \tag{6}$$

and following that components of $\boldsymbol{b}$ must be summable to unity, we also get the additional $(N-1)$ constraints:

$$b_1 \leq 1;\; b_2 \leq 1; ...\; ;\; b_{N-1} \leq 1 \tag{7}$$

with the performance maximization objective:

$$\max\{b_1 p_1 + b_2 p_2 + ... + b_N p_N\} \tag{8}$$

Note that (6) and (7) define N linear constraints to maximize the objective (8) with N basic values $\{b_1, b_2, ..., b_N\}$. Following the duality property of such convex problems [5], [13], we can also formulate the dual (and equivalent) problem that finds $\boldsymbol{b}$ for any specified performance target p_t. That is, appropriate biases can be found to meet p_t with the $(N-1)$ constraints of (7) and the additional constraint on expected performance:

$$b_1 p_1 + b_2 p_2 + ... + (1 - \sum_{n=1}^{N-1} b_n) p_N = p_t \tag{9}$$

with the data cost minimization objective:

$$\min\{b_1 d_1 + b_2 d_2 + ... + b_N d_N\} \tag{10}$$

Thus, determining $\boldsymbol{b}$ is a convex problem that can be readily solved by any convex optimization technique [5], [6], [13], such as the simplex method [5], [6]. That is, an appropriate biasing value $\boldsymbol{b}$ to use for training can be found for any specified target data cost d_t by solving for $\boldsymbol{b}$ in (6)-(8), or any target on expected performance p_t by solving (7), (9), and (10).

E. Final Observations

In considering the performance of biased mixtures, the quality of expert selections from $\mathcal{E}$ is regulated by the complexity of the gating function $G(\boldsymbol{x}; \mathcal{W}_g)$; where increasing the complexity of $G(\boldsymbol{x}; \mathcal{W}_g)$ can improve selections (e.g., by increasing the number of learnable weights), albeit with diminishing returns. In addition, and in the case of bias enforcement, we intuitively expect the quality of selections to be directly correlated with batch sizes used for training. That is, low batch size settings may not expose gating functions to a sufficient amount of variance in inputs to make selections of benefit, and setting higher batch sizes is favorable.

Importantly, applications of biased mixtures allow gating functions $G(\boldsymbol{x}; \mathcal{W}_g)$ to wholly observe inputs $\boldsymbol{x}$ prior to selecting experts for data-economy. That is, biased mixtures can be distributed to allow for gating before preprocessing to produce sampled inputs $P_n(\boldsymbol{x})$, and before inputs are subsequently sent to remote models for visual inference (as illustrated in Fig. 2). As a result, the constraint for gating functions is not input size, but the processing capability onboard visual sensors. We also note that, expert selection methods proposed in this paper can be applied with mixtures comprising experts that are optimized for low data cost via additional task-specific dimensionality reduction methods, and experts that use different modalities to make their inferences (as illustrated in (c) of Fig. 2). Finally, while our work studies the problem of reducing data utility, $\boldsymbol{b}$ can also be specified to prioritize any other expert property whenever constraints are properly quantified and made available to the proposed gating architecture (e.g., to meet constraints on power consumption or latency).

IV. Evaluation

A. Benchmarks and Evaluation Method

To show how biased mixtures can optimize data costs of inference for different problems, we evaluate on three computer vision tasks: *(i)* object detection, *(ii)* image super resolution, and *(iii)* realtime action classification. In reporting results for all tasks, we compare our method against two alternatives:

1) *Previously Proposed Models*: To benchmark our results against relevant task-specific solutions, we consider the performance of constituent experts when optimized for different data cost constraints. In biased mixtures, this corresponds to specifying $\boldsymbol{b}$ as a one-hot vector, and measures performance when the same amount of data is used for all inputs during inference (e.g., when $\boldsymbol{b} = [0, 1, 0]$ only E_2 is used for inference). We report this to benchmark against previous work and to highlight the benefit of uniquely dividing the input space for each expert.
2) *Random Selection*: Here, experts are randomly selected for inference at test time in order to satisfy the model biasing requirement $\boldsymbol{b}$. This is to serve as the lower bound of performance when biased mixtures are used and the specified expert utility bias is met.

Importantly, when considering the problem of task-agnostic model optimization under data cost constraints, there is no previous work similar to ours (see Section II). That is why, we benchmark against the maximum performance achievable by recently proposed *task-specific* solutions when their input volumes are adjusted to meet different constraints on data cost. That is, *biased mixtures consist of experts that also stand in as external benchmarks*. To highlight the latter, benchmark results of constituent experts are indicated in comparative plots by markers on dotted lines.

For clarity, and to ensure consistency of representation across all tasks, we report the per input data cost of inference $\bar{d}$ as the average amount of data seen by the mixture after inputs are fully decompressed. For each evaluated task we specify how the data cost for each expert d_n is measured (i.e., the data cost associated with $P_n(\boldsymbol{x})$). For a concise measure of how well models preform across different specified data cost constraints of $d_t \in [d_{\min}, d_{\max}]$, and with $p_{\text{test}}(d_t)$ denoting test performance when the target data cost is d_t, we report the area under curve when data cost is normalized as:

$$\rho = \int_0^1 p_{\text{test}}(d_{\min} + t(d_{\max} - d_{\min}))\, dt \tag{11}$$

For all mixtures, we specify the gating model (i.e., $f(\boldsymbol{x}; \mathcal{W}_g)$) as a single conv-pool layer followed by a fully connected network. To ensure that the model selection process is of low complexity for all tasks, we use ReLU activated depthwise separable convolutions [43], and report the per input number of multiply-accumulate gating operations C_g. We use cross-validation to optimize the biasing weight w_{bias} and report the best performance when soft regularization is used. After all experts included in the mixture are individually optimized, biased mixtures are trained by updating the weights of the

gating function exclusively, and the weights of experts are not fine-tuned further. We have found that using higher batch sizes is helpful when training biased mixtures, because it exposes the mixture to a more varied set of input examples to partition to each expert meaningfully. Therefore, to ensure gating functions learn meaningful features for batch partitioning, for all tasks we set the batch size to 128 and the learning rate to 10^{-4}.

B. Single-Shot Object Detection

We test our method on single-shot detection (SSD) to reduce the data requirement for object detection while maintaining high accuracy. Recent work [19], [20] [51] showed that SSD models vary widely in performance and complexity when input sizes are adjusted. When considering the varying degrees of complexity of natural images, we expect that the minimum required subsampling rate of inputs for accurate object detection should vary accordingly. To demonstrate this, we train a biased mixture of experts where each expert is optimized for a different image subsampling rate, and use the recent work of Liu *et. al* [25] as a baseline for all experts (for an illustration, see (a) of Figure 2). When the resolution of inputs to each expert is $R_n \times R_n$ pixels, we measure the data cost associated with $P_n(\boldsymbol{x})$ as $3 \times R_n \times R_n \times K$, where 3 is the number of color channels in RGB inputs, and K is the number of bytes needed to store floating point decimals.

We use VGG16 [12] and ResNet50 [17] for feature extraction and evaluate all models using 300 regional proposal boxes for VGG16 [12], and 50 regional proposal boxes for ResNet50 [17]. Following recent work [20], [25], we train on COCO training data while excluding the 8k mini-eval images used in the 2012 challenge [24], and report performance as the mean Average Precision (mAP) on COCO (07+12). We train mixtures for 20k steps to show our results when using soft regularization and bias enforcement, and in Table IV we detail the types and complexities of all layers used in devising the gating model $f(\boldsymbol{x}; \mathcal{W}_g)$. Inputs to the gating model are pre-processed as 224×224 center crops of 300×300 images, and we ensure that the gating complexity of all mixtures remains at $C_g < 10^8$ Mult-Add operations.

TABLE I: Single shot detection comparison on COCO [24] of biased mixtures of SSD [25] experts against other benchmarks. Resolutions $\{R_n\}$ and data costs $\{d_n\}$ are reported for all experts.

$\{R_n\}$= $\{100, 150, 300\}$(Pixels); $\{d_n\} = \{120, 270, 1080\}(kB)$

Feature Extractor	Biasing Method	mAP(d_t) (%) when d_t = d_{max}	$\frac{d_{max}}{2}$	$\frac{d_{max}}{3}$	ρ
VGG16 [44]	Benchmark Experts [44]	80.0	70.0	66.7	70.9
	Proposed $\boldsymbol{b}$ Enforcement		**72.5**	**70.9**	**73.1**
	Soft Regularization [41]		67.1	65.0	68.9
	Random Selection		66.3	63.4	68.2
ResNet50 [17]	Benchmark Experts [17]	75.7	65.1	61.3	66.1
	Proposed $\boldsymbol{b}$ Enforcement		**67.8**	**65.9**	**68.3**
	Soft Regularization [41]		62.2	59.9	64.2
	Random Selection		61.9	57.4	63.3

Figure 4 shows the relationship between imposed bias, data cost, and mAP when three VGG16 experts are used for single shot detection, where the resolution of inputs to each expert is $\{R_n\} = \{100, 150, 300\}$. Notably, biased mixtures optimized with bias enforcement provide the slowest degradation in mAP for lower data costs, with diminishing gains when more data is available at test time. Specifically, biasing via enforcement outperforms individual experts by 7.5% when an average of 220 kilobytes per image is allowed, which is equal to the performance of individual experts at 490 kilobytes. That is, when the minimum acceptable mAP is 70%, a reduction of 270 kilobytes in required data is achieved by our proposal (which is equivalent to a saving of 55% in bitrate).

TABLE II: mAP performance of individual experts over their assigned input examples as determined by $G(\boldsymbol{x}; \mathcal{W}_g)$. Baseline expert accuracies before gating are reported in $\{E_n$ mAP$\}$, and are measured as the accuracy of each expert over all COCO inputs [24]. Values in parentheses show differences relative to expert baseline accuracies in $\{E_n$ mAP$\}$.

$\{R_n\}$= $\{100, 150, 300\}$; $\{E_n$ mAP$\}$=$\{57.91, 64.60, 80.01\}$ (%)

Bias Enforcement mAP (%)				
$\boldsymbol{b}$	E_1	E_2	E_3	BMoE
$[0.8, 0.1, 0.1]$	70.62 (+12.72)	60.07(−4.52)	72.02(−7.99)	69.71
$[0.5, 0.3, 0.2]$	76.21 (+18.31)	61.37(−3.23)	73.60(−6.41)	71.24
$[0.2, 0.3, 0.5]$	78.53 (+20.63)	63.31(−1.29)	77.59(−2.40)	73.50
Soft Regularization mAP (%)				
$\boldsymbol{b}$	E_1	E_2	E_3	BMoE
$[0.8, 0.1, 0.1]$	60.73 (+2.83)	60.72(−3.87)	73.61(−6.40)	62.02
$[0.5, 0.3, 0.2]$	63.19 (+5.31)	62.01(−2.58)	76.03(−4.01)	65.41
$[0.2, 0.3, 0.5]$	56.95 (−0.94)	62.66(−1.93)	76.80(−3.20)	68.59

TABLE III: Relation between gating complexity, batch size, and performance when bias enforcement is used.

C_g (Mult-Adds)	M	ρ when $\{R_n\}$ = $\{100, 300\}$(Pixels) VGG16 [44]	$\{100, 300\}$(Pixels) ResNet [17]	$\{100, 150, 300\}$(Pixels) VGG16 [44]	$\{100, 150, 300\}$(Pixels) ResNet50 [17]
23,048,576	16	68.40	64.11	69.27	64.46
	32	70.35	65.89	70.25	65.57
	64	**70.93**	**66.24**	**71.16**	**65.72**
26,194,304	16	70.85	66.92	71.82	67.04
	32	71.49	67.25	72.50	67.41
	64	**71.84**	**67.59**	**72.97**	**68.04**
38,700,216	16	70.93	67.01	72.10	67.33
	32	71.58	67.25	73.07	68.26
	64	**71.86**	**67.62**	**73.13**	**68.30**

To further assess how biased mixtures learn useful bifurcations of input space, Table II details the performance of each expert on their assigned subset of inputs. Notably, Table II highlights how easier input examples are passed to the data-efficient expert E_1, resulting in increased accuracies of E_1 compared to its baseline accuracy of 57.91% which is measured over all inputs. Conversely, biased mixtures pass more difficult examples to E_2 and E_3, resulting in lower accuracies over their assigned inputs compared to their baseline accuracies. Interestingly, and especially for bias enforcement, Table II also shows how improved accuracies of E_1 (which correlate with how "easy" its assigned inputs are to classify) are inversely proportional to the number of examples passed to it, as reflected by b_1 (e.g., a difference of $+12.72$ percentile points in accuracy when $b_1 = 0.8$, compared to an increase of $+18.31$ when $b_1 = 0.5$).

In Table I we show the performance of biased mixtures when applied to multiple models, and report ρ as a comprehensive measure of model performance across data costs. When compared to random selection, we note that for both ResNet50 [17] and VGG16 [44], imposing bias on mixtures provides the

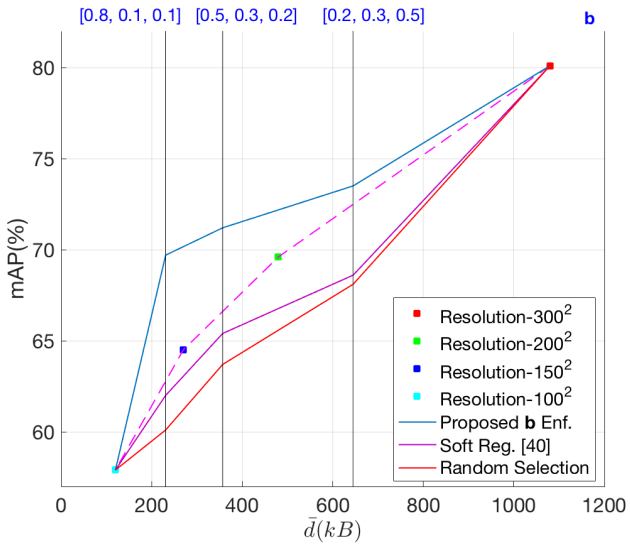


Fig. 4: Single shot detection performance comparison of biased mixtures of VGG16 [44] experts against other benchmarks when $\{R_n\} = \{100, 150, 300\}$. The performance of individual experts is shown on the dotted line.

highest gain when lower values of data cost are considered (e.g., when $d_t < \frac{d_{max}}{3}$). Compared to soft regularization, and for all mixture configurations, we found that bias enforcement is a much more effective method for training biased mixtures (this is also true for all other tasks evaluated). We hypothesise this is because, when bias enforcement is used only the task loss is back-propagated during training, which causes less competition between losses and therefore less local minima to exist in solution space.

TABLE IV: Layer complexities C of the gating model $f(\boldsymbol{x}; \mathcal{W}_g)$ for biased mixtures evaluated on single shot detection. Expert input resolutions are specified as $\{R_n\} = \{100, 150, 300\}$ and $N = 3$.

Layer Type	Filter Shape	Stride	Input Shape	C (Mult-Adds)
Convolutional	$3 \times 3 \times 3 \times 64$	2	$224 \times 224 \times 3$	$2,747,136$
Avg. Pooling	7×7	5	$111 \times 111 \times 64$	—
Flatten Op.	–	–	$21 \times 21 \times 64$	—
Fully Connected	28224×1024	–	1×28224	$28,901,376$
Fully Connected	1024×3	–	1×1024	3072

In Table III we study the effect of adjusting the gating complexity C_g, batch size M, and number of experts N on the performance of biased mixtures when bias enforcement is used. When we consider all mixtures, we find that batch size is critical to performance. This is because bias is enforced on a per batch basis, and to make meaningful decisions the gating function needs to be exposed to an ample amount of variance between examples. We also see that increasing the complexity of gating does increase performance by helping partition the input space more effectively. However, this effect saturates at $C_g \approx 3.8 \times 10^7$ Mult-Add operations, which demonstrates that the optimal hyperplane to partition input space for $N \leq 3$ experts can be learned with low complexity.

By comparing the left and right part of Table III, we see that adding more experts to the mixture provides a modest increase to performance. This is because having more experts allows the mixture to further exploit the variance in different input sub-spaces (if any such variance exists). To see the extent to which this is true, in Figure 5 we adjust the limits of allowed input resolutions to the mixture R_{min} and R_{max}, and report ρ when considering different values of N. Importantly, we see that when the difference between R_{min} and R_{max} is lower, using more experts yields less gain in performance, to the point where using more than three experts for $(R_{min}, R_{max}) = (100, 300)$ does not provide any benefit. This is because, while setting high values of N increases the number of intermediate resolutions between R_{min} and R_{max}, the difference $(R_{max} - R_{min})$ correlates with the amount of discernable adequacy between experts, which in turn correlates with the benefit of including more experts.

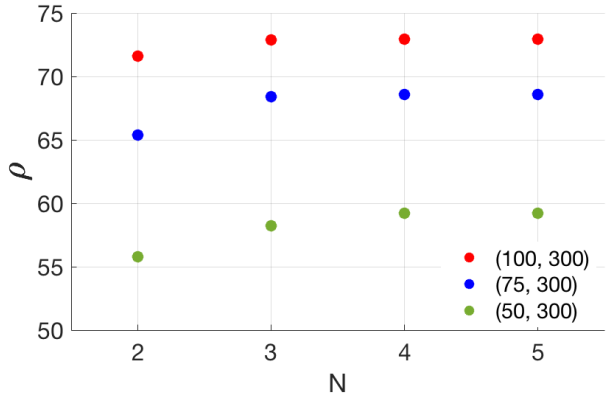


Fig. 5: ρ when bias enforcement is used and the number of experts N is configured. VGG16 [44] is used for feature extraction, and different colors indicate the resolution limits (R_{min}, R_{max}) allowed to the mixture (where N determines the number of intermediate input resolutions included).

C. Image Super-Resolution

We test the applicability of biased mixtures on Single Image Super resolution (SISR), an image reconstruction task where spatial features of high-resolution images are inferred from low-resolution input images. Several recent proposals have shown good performance in terms of image reconstruction accuracy and computational efficiency [11], [42], [47] [54]. However, current super resolution models do not take into account the variable amount of high-frequency edge content between images. That is, when reconstructing images which contain many high frequency elements, SISR models are likely to benefit from higher resolution input images, while images comprising predominately low-frequency content can be inferred just as well from lower resolution inputs. This is true also when considering different parts of an image, which usually vary in the breadth of their frequency elements. To demonstrate this, we train biased mixtures to determine the needed input resolution for good image reconstruction, and in doing so, we show how different image parts can be adaptively upsampled subject to their content. Such decisions about selected super-resolution experts can also be augmented

to existing media streaming standards (e.g., DASH/HLS in HTTP [19]) for adaptive subsampling prior to transmission.

We evaluate on the NTIRE17 challenge dataset DIV2K [1], and to expose biased mixtures to the intra-image variance of frequency elements, images are divided using a fixed grid into parts of size 64×64 pixels, and super-resolution is performed on each part separately (for an illustration, see (b) of Figure 2). By inspecting the low-level semantics of each image part, the mixture selects the most data efficient expert for reconstruction to preform an upscaling from the set $\{S_n\} = \{\times 4, \times 3, \times 2\}$. For each expert that upscales inputs with a factor of S_n to match the target resolution of 64×64 pixels, we measure the associated data cost as $d_n = (64/S_n)^2 \times K$, where K is the number of bytes needed to store floating point decimals. To expose gating to the high frequency components of input images, inputs to the gating model are not subsampled, and are maintained at the original resolution of resolution of 64×64 pixels. For all biased mixture results, mixtures are trained for 20 epochs and we ensure the complexity of the gating function is set to $C_g < 10^7$ Mult-Add operations.

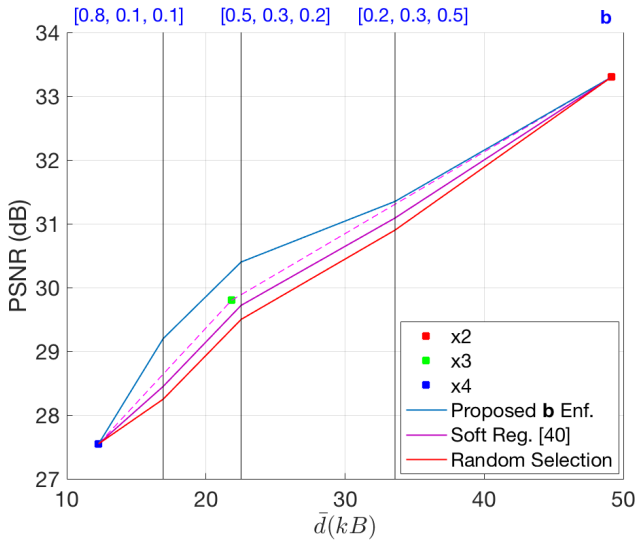


Fig. 6: Super resolution performance comparison of biased mixture of ESPCN [42] experts and other benchmarks when $\{S_n\} = \{\times 4, \times 3, \times 2\}$.

TABLE V: Image super resolution comparison on DIV2K [1] of biased mixtures and other benchmarks. Upscale factors $\{S_n\}$ and data costs $\{d_n\}$ are reported for all experts.

$\{S_n\} = \{\times 4, \times 3, \times 2\}$; $\{d_n\} = \{13.9, 21.8, 49.2\}(kB)$

Model	Biasing Method	PSNR(d_t) (dB) when $d_t =$ d_{max}	$\frac{d_{max}}{2}$	$\frac{d_{max}}{3}$	ρ
ESPCN [42]	Benchmark Experts [42]	33.3	30.4	28.4	30.7
	Proposed ***b*** Enforcement		**30.7**	**28.8**	**31.0**
	Soft Regularization [41]		30.0	28.1	30.6
	Random Selection		29.8	28.0	30.5
F-SRCNN [11]	Benchmark Experts [11]	32.8	29.8	28.0	30.3
	Proposed ***b*** Enforcement		**30.1**	**28.3**	**30.5**
	Soft Regularization [41]		29.3	27.6	30.1
	Random Selection		29.2	27.5	30.0

In Table V we compare biased mixtures against other benchmarks when using ESPCN [42] and FRSCNN [11] as baselines, in Table VI we detail the performance of experts over their assigned subsets of input, and in Figure 6 we show the relationship between average data cost and PSNR when considering ESPCN [42]. Notably from Figure 6, when bias enforcement is used and $\bar{d}$ is within the range of 18-22 kilobytes, biased mixtures outperform single experts with an average difference of 0.4 dB. Over the same range of values of $\bar{d}$, and when compared to random selection, bias enforcement provides an average improvement of 0.7 dB. This highlights the magnitude of intra-image high variance in required input resolution for image reconstruction, which is not considered by random selection and optimized experts. Overall, Figure 6 and Table V show that biased mixtures outperform individual experts most when $\bar{d} < 20$ kilobytes, with diminishing gains in performance for higher values of $\bar{d}$.

TABLE VI: PSNR performance of individual experts over their assigned input examples as determined by $G(\boldsymbol{x}; \mathcal{W}_g)$. Baseline PSNR values before gating are reported in $\{E_n$ PSNR$\}$, and are measured for each expert over all DIV2K inputs [1]. Values in parentheses show differences relative to baseline reconstruction accuracies in $\{E_n$ PSNR$\}$.

$\{S_n\} = \{\times 4, \times 3, \times 2\}$;$\{E_n$ PSNR$\}$=$\{27.61, 29.86, 33.31\}$ (dB)

Bias Enforcement PSNR (dB)				
b	E_1	E_2	E_3	BMoE
$[0.8, 0.1, 0.1]$	28.97 (+1.36)	28.96(−0.89)	32.35(−1.02)	29.31
$[0.5, 0.3, 0.2]$	30.14 (+2.53)	29.62(−0.23)	32.64(−0.67)	30.49
$[0.2, 0.3, 0.5]$	30.09 (+2.48)	29.74(−0.14)	32.97(−0.33)	31.42
Soft Regularization PSNR (dB)				
b	E_1	E_2	E_3	BMoE
$[0.8, 0.1, 0.1]$	27.95 (+0.34)	28.93(−0.92)	32.24(−1.06)	28.48
$[0.5, 0.3, 0.2]$	28.56 (+0.95)	29.71(−0.14)	32.57(−0.73)	29.71
$[0.2, 0.3, 0.5]$	28.51 (+0.90)	29.77(−0.08)	32.91(−0.39)	31.09

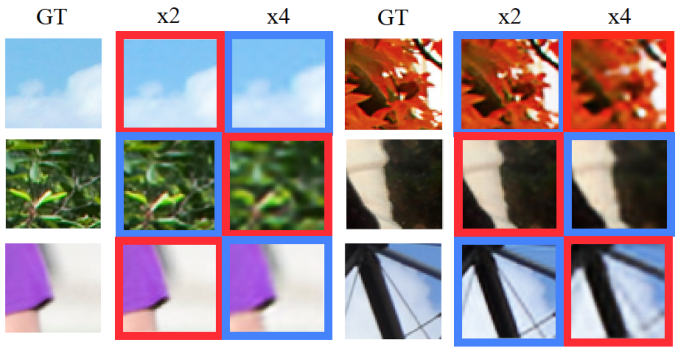


Fig. 7: Examples of expert assignments to different image parts. Selected and non-selected experts are respectively highlighted by blue and red borders. Note the exploitable variance in detail between images, which translates into the data cost savings reported in Table V.

In Figure 7 we show examples of expert selections made by the biased mixture to resolve different 64×64 inputs when bias enforcement is used. The mixture learns to pass image parts with high frequency components to the $\times 2$ SISR model, and passes other less demanding parts to the $\times 4$ model (which are blurrier, due to the lower frequency of their components).

TABLE VII: Layer complexities C of the gating model $f(\boldsymbol{x}; \mathcal{W}_g)$ for biased mixtures evaluated on single image super-resolution. Expert upscaling factors are specified as $\{S_n\} = \{\times 4, \times 3, \times 2\}$ and $N = 3$.

Layer Type	Filter Shape	Stride	Input Shape	C (Mult-Adds)
Convolutional	$3 \times 3 \times 3 \times 64$	2	$64 \times 64 \times 3$	$224,256$
Avg. Pooling	3×3	2	$21 \times 21 \times 64$	—-
Flatten Op.	–	–	$10 \times 10 \times 64$	—-
Fully Connected	6400×512	–	1×6400	$3,276,800$
Fully Connected	512×3	–	1×512	$1,536$

D. Realtime Action Classification

We validate biased expert selection on *realtime* video action classification in the compressed domain. While the best performing action classification models operate on uncompressed video data, to reduce latency, the models proposed in recent work [8], [53] infer a low-resolution optical flow from codec motion vectors at high speeds for action classification. The classifiers of [8], [53] use two-stream architectures to infer actions, where spatial and temporal classifiers complement each other by learning different sets of features from their respective domains [40]. As such, for some action subsets, the use of only the temporal or spatial classifier can suffice in drawing accurate distinctions between actions, but combining the predictions of both provides the highest accuracy.

Distinct from other compute-exhaustive models for action classification [7], the recent proposals on *realtime* video classification [8], [53] use minimal volumes of data to ensure complexities and runtimes remain low. The work of [8], [53] also produces spatio-temporal modes directly from compressed bitstreams to bypass complexity overheads associated with dense optical flow estimation. We show how input volumes can be further reduced by learning which modes to use directly from data, by exposing only the spatio-temporal mode to gating functions that select which modes to send to remote realtime classification models [8], [53]. We do this such that all modalities (spatial and spatio-temporal) are sent to remote classifiers exclusively when videos are challenging to classify. Otherwise, only the temporal modalities are sent, thereby mitigating unnecessary traffic between sensors and remote classifiers (and we illustrate this in (c) of Figure 2).

We evaluate on UCF-101 [45] and measure the cost associated with the spatial mode as $F_s \times H_s \times W_s \times K \times 3$, where $F_s = 2$ is the number of RGB frames used, $H_s = 360$ and $W_s = 240$ are the height and width of inputs, and $K = 32$ is the number of bytes to store floating point decimals. For the temporal model, we measure the data cost as $F_t \times H_t \times W_t \times K \times 2$, where $H_t = 24$ and $W_t = 24$ are the height and width of approximated optical flow, and $F_t = 150$ is the number of frames used (two channels are used in optical flow to represent vertical and horizontal motion). Importantly, we select sampling rates akin to those of [8] which sets $F_s = 1$, $F_t \geq 10$, and the proposal of [53] which sets $F_s = 1$, $F_t \geq 100$. This is to meet complexity limits for realtime inference, where the benchmark models [8], [53] set modest sampling rates compared to other exhaustive methods [7], which typically use dense optical flow approximations with $F_s \geq 50$ and $F_t \geq 150$. Moreover, in implementing the benchmark model of Zhang *et al.* [53], we follow their method of upsampling 24×24 optical flow crops to 224×224 temporal mode inputs. However, upsampling is performed after inputs are sent via the $(*)$ operator of Fig. 2 (c), and therefore shape parameters remain at $H_t = 24$ and $W_t = 24$ when measuring data cost.

The fusion classifier uses both modalities to predict actions and is the most accurate, but requires a data cost equal the sum of both modalities. We include all modalities to train a mixture of experts $\{\text{Mode}_n\} = \{\text{Temporal}, \text{Spatial}, \text{Fusion}\}$, and train a gating function to select the most suitable modality to use for each input. Importantly, and to allow for lower complexities of gating, inputs to the gating model include only the temporal modes of videos, and spatial modes are not used. For all biased mixtures, we train for 80k steps and restrict the complexity of the gating function to $C_g < 10^8$ Mult-Add operations, where we detail the layer-wise complexities of gating in Table X.

TABLE VIII: Realtime action classification on UCF-101 [45] of biased mixtures of experts and other benchmarks. Modalities $\{\text{Mode}_n\}$ and data costs $\{d_n\}$ are reported for all experts.

$\{\text{Mode}_n\} = \{\text{Temporal}, \text{Spatial}, \text{Fusion}\}$; $\{d_n\} = \{737.3, 1843.0, 2580.5\}(kB)$

Model	Biasing Method	Accuracy(d_t) (%) when $d_t =$ d_{max}	$\frac{d_{max}}{2}$	$\frac{d_{max}}{3}$	ρ
MV-3DCNN [8]	Benchmark Experts [8]	88.0	79.0	77.9	80.9
	Proposed $\boldsymbol{b}$ Enforcement		**82.0**	**80.4**	**83.5**
	Soft Regularization [41]		80.3	78.0	81.9
	Random Selection		78.8	77.3	81.3
EMV-CNN [53]	Benchmark Experts [53]	85.6	76.6	75.5	78.7
	Proposed $\boldsymbol{b}$ Enforcement		**80.2**	**79.2**	**81.3**
	Soft Regularization [41]		77.2	75.6	79.7
	Random Selection		75.7	74.9	79.0

TABLE IX: Accuracy of individual experts over their assigned inputs as determined by $G(\boldsymbol{x}; \mathcal{W}_g)$. Baseline accuracies before gating are reported in $\{E_n$ Acc.$\}$, and are measured for each expert over all UCF-101 inputs [45]. Differences relative to expert baseline accuracies in $\{E_n$ Acc.$\}$ are shown in parentheses.

$\{\text{Mode}_n\} = \{\text{Temporal}, \text{Spatial}, \text{Fusion}\}$; $\{E_n$ Acc.$\} = \{77.84, 80.11, 88.03\}$ (%)

Bias Enforcement Accuracy (%)				
$\boldsymbol{b}$	E_1	E_2	E_3	BMoE
$[0.8, 0.1, 0.1]$	81.80 (+3.97)	78.99(−1.12)	84.51(−3.52)	81.8
$[0.5, 0.3, 0.2]$	83.65 (+5.81)	79.34(−0.76)	85.34(−2.68)	82.7
$[0.2, 0.3, 0.5]$	83.44 (+5.60)	79.38(−0.72)	85.38(−2.64)	83.2
Soft Regularization Accuracy (%)				
$\boldsymbol{b}$	E_1	E_2	E_3	BMoE
$[0.8, 0.1, 0.1]$	78.41 (+0.57)	78.10(−2.01)	84.59(−3.43)	79.00
$[0.5, 0.3, 0.2]$	81.04 (+3.20)	78.51(−1.61)	85.13(−2.90)	81.10
$[0.2, 0.3, 0.5]$	78.74 (+0.91)	78.66(−1.44)	85.30(−2.72)	80.02

TABLE X: Layer complexities C of the gating model $f(\boldsymbol{x}; \mathcal{W}_g)$ for biased mixtures evaluated on realtime action classification. Expert modalities are specified as $\{\text{Mode}_n\} = \{\text{Temporal}, \text{Spatial}, \text{Fusion}\}$ and $N = 3$. Note that the gating model $f(\boldsymbol{x}; \mathcal{W}_g)$ ingests only temporal modalities of $\boldsymbol{x}$.

Layer Type	Filter Shape	Stride	Input Shape	C (Mult-Adds)
Convolutional	$3 \times 3 \times 320 \times 64$	2	$24 \times 24 \times 320$	$3,363,840$
Flatten Op.	–	–	$11 \times 11 \times 64$	—
Fully Connected	7744×1024	–	1×7744	$7,929,856$
Fully Connected	1024×3	–	1×1024	$3,072$

In Table VIII we compare the performance of biased mixtures against other benchmarks when using the spatial and temporal classifiers of [8] and [53] as baselines, and in Table IX we detail the performance of experts over their assigned input subsets as determined by $G(\boldsymbol{x}; \mathcal{W}_g)$. From Table VIII, we first note that both biasing methods outperform random selection, by up to 1% for soft regularization and up to 3.8% for bias enforcement. This indicates that the biased mixture learns to discern confusing classes for particular modalities to pass them to others. Notably, when $\bar{d} = \frac{d_{max}}{3} = 860$ kilobytes, bias enforcement gives an accuracy 1.4% higher than that of the optimized experts at $\frac{d_{max}}{2} = 1290$ kilobytes, which requires 430 kilobytes more in data cost.

In Figure 8 we show the relationship between $\bar{d}$ and action classification accuracy for instances of $\boldsymbol{b}$ when biased mixtures of MV-3DCNN [8] experts are used and the mode of each expert is $\{\text{Mode}_n\} = \{\text{Temporal}, \text{Spatial}, \text{Fusion}\}$. We first note that, due to the low resolution of its inputs, the temporal classifier requires the least amount of data and can predict actions with an accuracy of 77.8%. By selecting among the three modes, both biasing methods outperform random selection, with bias enforcement increasing accuracy by up to 3.4% for when $\bar{d} = 1032$ kilobytes. Notably, and when using the temporal classifier for 80% of videos at $\bar{d} = 1032$ kilobytes (i.e., when $\boldsymbol{b} = [0.8, 0.1, 0.1]$), bias enforcement is 1.6% more accurate than the spatial classifier (which requires 811 kilobytes more in data, equivalent to an increase of 78% in data cost). The latter shows the extent to which biased mixtures can improve performance by using modest amounts of data, even compared to individual models that require substantially more in data cost. Moreover, Table IX highlights how inputs are appropriately passed to experts for data-economic classification. Specifically, it shows how biased mixtures learn to use the data-efficient temporal model for inputs that are easier to classify, where temporal modalities are likely to suffice for accurate classification. For example, this is evident when $b_1 = 0.5$ and $b_1 = 0.2$, where the temporal classifier E_1 respectively gains $+5.81$ and $+5.60$ percentile points in classifying its assigned inputs when compared to its baseline accuracy measured over all videos of UCF-101 [45]. On the other hand, Table IX also shows how more difficult inputs are passed to the spatial and fusion classifiers, resulting in a modest loss of accuracy when classifying their assigned inputs.

To visualize how different modalities are assigned to videos,

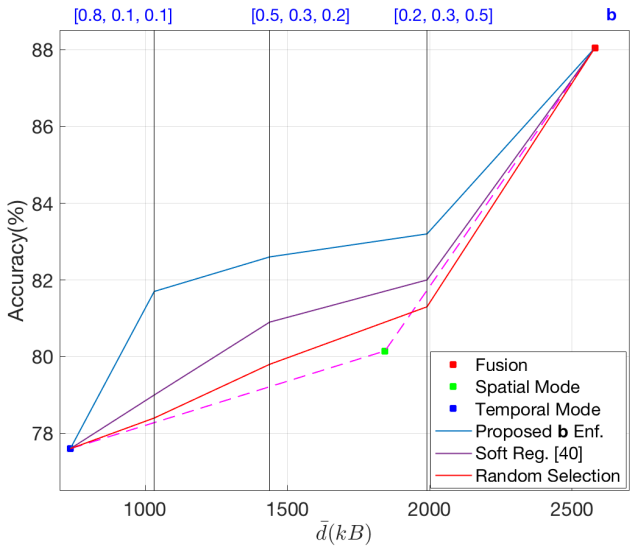
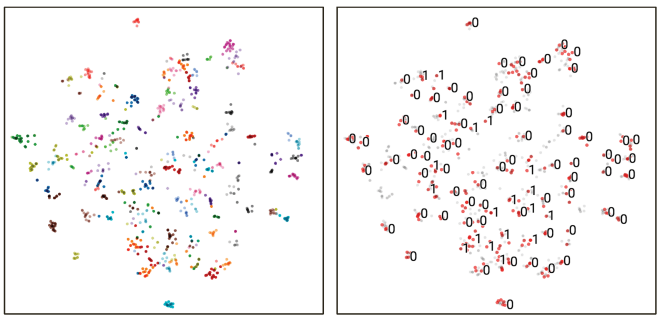


Fig. 8: Realtime action classification performance comparison of biased mixtures of MV-3DCNN [8] experts, with expert modalities $\{\text{Mode}_n\}$={Temporal,Spatial, Fusion}.



(a) (b)

Fig. 9: t-SNE [27] projections of 1024 UCF101 videos, where in (a) colours indicate different classes, and (b) mode assignments are shown as 0 or 1 for the temporal and fusion classifiers respectively. Zoom in to view in high-resolution.

in Figure 9 we show two-dimensional t-SNE [27] projections of 1024 UCF101 examples as embedded by the last layer of the temporal classifier. For clarity of presentation, we use a biased mixture of two modalities $\{\text{Mode}_n\} = \{\text{Temporal}, \text{Fusion}\}$ and set $\boldsymbol{b} = [0.75, 0.25]$. In this way, we show the relation between different class labels and assigned modalities. Notably in Fig. 9 (a), the middle region highlights instances of different classes which are more entangled and therefore harder to classify. Moreover, we observed that temporal modes of inputs are typically more difficult to discern when they contain: (i) significant camera movement, leading to noisier motion flow, or (ii) relatively static scenes, resulting in sparse optical flow approximations. For a sample of instances, Fig. 9 (b) shows modalities selected by the biased mixture for action classification. It can be seen from Fig. 9 that the biased mixture learns to favor using the temporal classifier for video clusters that are comparatively isolated, and easy to discern from other clusters. Conversely, when video instances are not clearly

clustered or isolated (mostly located in the middle), the biased mixture selects the fusion model. In other words, the biased mixture tends to select the data-exhaustive fusion model when videos are harder to classify (as indicated by label 1 in (b) of Fig. 9), and temporal modes are exclusively used for inputs sufficiently discernable from only temporal representations (as indicated by label 0 in (b) of Fig. 9). Hence, Fig. 9 shows how biased expert mixtures can find useful bifurcations of input space such that only necessary modalities are used for action classification, and less data is used whenever possible.

V. Conclusion

We introduce biased expert utility in mixtures of experts for effective partitioning of input space to meet constraints on data availability at test time. We propose two methods for training biased mixtures, and evaluate their performance on multiple models for all investigated tasks. We show how biased mixtures are applicable to any situation wherein experts vary in data requirement and performance, and demonstrate this on a wide range of computer vision tasks. Our validation shows that, especially for lower ranges of allowed data cost, biased mixtures significantly outperform baseline models optimized to meet the same constraints on available data. We also show how useful gating inferences that prioritise data economy can be realized with complexities that do not exceed 10^8 Multi-Add operations, which are feasible to run even on embedded computation units (e.g., ARM Cortex-M7). Within contexts of distributed visual inference, and to meet different constraints on data transfer and bandwidth at test time, all of our observations and tests show the importance of conditioning data utility for visual inference to the local proximities and properties of inputs within their space. In other words, the importance of doing so is applicable to all presented vision tasks, and is likely to extend to other visual inference tasks in order to mitigate unnecessary burdens on communication resources and sensor hardware. We finally note that an important advantage of biased mixtures is the flexibility at which they can be applied, in that, biased mixtures *do not modify* their constituent experts, but rather *augment* their function with an input preprocessing stage that allows for data economy in inference.

References

[1] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 126–135.

[2] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," *arXiv preprint arXiv:1611.01704*, 2016.

[3] E. Bengio, P.-L. Bacon, J. Pineau, and D. Precup, "Conditional computation in neural networks for faster models," *arXiv preprint arXiv:1511.06297*, 2015.

[4] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[5] K. H. Borgwardt, *The Simplex Method: a probabilistic analysis*. Springer Science & Business Media, 2012, vol. 1.

[6] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[7] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 4724–4733.

[8] A. Chadha, A. Abbas, and Y. Andreopoulos, "Video classification with cnns: Using the codec as a spatio-temporal activity sensor," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 475–485, 2017.

[9] S.-P. Chuah, N.-M. Cheung, and C. Yuen, "Layered coding for mobile cloud gaming using scalable blinn-phong lighting," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3112–3125, 2016.

[10] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486–1494.

[11] C. Dong, C. C. Loy, and X. Tang, "Accelerating the super-resolution convolutional neural network," in *European Conference on Computer Vision*. Springer, 2016, pp. 391–407.

[12] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1933–1941.

[13] M. Fiedler, J. Nedoma, J. Ramík, J. Rohn, and K. Zimmermann, *Linear optimization problems with inexact data*. Springer Science & Business Media, 2006.

[14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[15] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[16] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Advances in neural information processing systems*, 2015, pp. 1135–1143.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[20] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *IEEE CVPR*, vol. 4, 2017.

[21] M. Jubran, A. Abbas, A. Chadha, and Y. Andreopoulos, "Rate-accuracy trade-off in video classification with deep convolutional neural networks," *IEEE Transactions on Circuits and Systems for Video Technology*, 2018.

[22] Y. Li, D. Liu, H. Li, L. Li, Z. Li, and F. Wu, "Learning a convolutional neural network for image compact-resolution," *IEEE Transactions on Image Processing*, vol. 28, no. 3, pp. 1092–1107, 2019.

[23] J. Lin, Y. Rao, J. Lu, and J. Zhou, "Runtime neural pruning," in *Advances in Neural Information Processing Systems*, 2017, pp. 2181–2191.

[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[25] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[26] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Advances in Neural Information Processing Systems*, 2018, pp. 1640–1650.

[27] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[28] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015.

[29] J. Martin, Y. Fu, N. Wourms, and T. Shaw, "Characterizing netflix bandwidth consumption," in *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*. IEEE, 2013, pp. 230–235.

[30] F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. V. Gool, "Practical full resolution learned lossless image compression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 629–10 638.

[31] L. Mescheder, S. Nowozin, and A. Geiger, "Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2391–2400.

[32] J. Miano, *Compressed image file formats: Jpeg, png, gif, xbm, bmp*. Addison-Wesley Professional, 1999.
[33] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 771–10 780.
[34] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," *arXiv preprint arXiv:1601.06759*, 2016.
[35] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.
[36] G. Roelofs and R. Koman, *PNG: the definitive guide*. O'Reilly & Associates, Inc., 1999.
[37] J. T. Rolfe, "Discrete variational autoencoders," *arXiv preprint arXiv:1609.02200*, 2016.
[38] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications," *arXiv preprint arXiv:1701.05517*, 2017.
[39] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa, "Generate to adapt: Aligning domains using generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8503–8512.
[40] L. Sevilla-Lara, Y. Liao, F. Guney, V. Jampani, A. Geiger, and M. J. Black, "On the integration of optical flow and action recognition," *arXiv preprint arXiv:1712.08416*, 2017.
[41] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean, "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer," *arXiv preprint arXiv:1701.06538*, 2017.
[42] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.
[43] L. Sifre and S. Mallat, "Rigid-motion scattering for texture classification," *arXiv preprint arXiv:1403.1687*, 2014.
[44] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[45] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.
[46] S. Srivastava and B. Lall, "Superresolution based medical image compression for mobile platforms," in *Workshop on Machine Learning for HealthCare*, 2015.
[47] R. Timofte, E. Agustsson, L. Van Gool, M.-H. Yang, and L. Zhang, "Ntire 2017 challenge on single image super-resolution: Methods and results," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 114–125.
[48] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7167–7176.
[49] A. Van den Oord, N. Kalchbrenner, L. Espeholt, O. Vinyals, A. Graves *et al.*, "Conditional image generation with pixelcnn decoders," in *Advances in neural information processing systems*, 2016, pp. 4790–4798.
[50] W. Wang, Y. Huang, Y. Wang, and L. Wang, "Generalized autoencoder: A neural network framework for dimensionality reduction," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 490–497.
[51] W. Wang, J. Shen, and L. Shao, "Video salient object detection via fully convolutional networks," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 38–49, 2018.
[52] W. K. Wong, Z. Lai, J. Wen, X. Fang, and Y. Lu, "Low-rank embedding for robust image feature extraction," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 2905–2917, 2017.
[53] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time action recognition with deeply transferred motion vector cnns," *IEEE Transactions on Image Processing*, vol. 27, no. 5, pp. 2326–2339, 2018.
[54] Y. Zhang, Q. Fan, F. Bao, Y. Liu, and C. Zhang, "Single-image super-resolution based on rational fractal interpolation," *IEEE Transactions on Image Processing*, vol. 27, no. 8, pp. 3782–3797, 2018.
[55] L. F. W. Z. Zhaoyang Zhang, Zhanghui Kuang, "Temporal sequence distillation: Towards few-frame action recognition in videos," in *Arxiv: 1808.05085*, 2018.