Combining Modular Skills in Multitask Learning

Edoardo M. Ponti[μ,γ] **Alessandro Sordoni**[σ] **Yoshua Bengio**[μ,ρ] **Siva Reddy**[μ,γ]
[μ]Mila – Quebec AI Institute [γ]McGill University
[σ]Microsoft Research Montréal [ρ]Université de Montréal
[μ]{edoardo-maria.ponti,yoshua.bengio,siva.reddy}@mila.quebec
[σ]alsordon@microsoft.com

Abstract

A modular design encourages neural models to disentangle and recombine different facets of knowledge to generalise more systematically to new tasks. In this work, we assume that each task is associated with a subset of latent discrete skills from a (potentially small) inventory. In turn, skills correspond to parameter-efficient (sparse / low-rank) model parameterisations. By jointly learning these and a task–skill allocation matrix, the network for each task is instantiated as the average of the parameters of active skills. To favour non-trivial soft partitions of skills across tasks, we experiment with a series of inductive biases, such as an Indian Buffet Process prior and a two-speed learning rate. We evaluate our latent-skill model on two main settings: 1) multitask reinforcement learning for grounded instruction following on 8 levels of the BabyAI platform; and 2) few-shot adaptation of pre-trained text-to-text generative models on CrossFit, a benchmark comprising 160 NLP tasks. We find that the modular design of a network significantly increases sample-efficiency in reinforcement learning and few-shot generalisation in supervised learning, compared to baselines with fully shared, task-specific, or conditionally generated parameters where knowledge is entangled across tasks. In addition, we show how discrete skills help interpretability, as they yield an explicit hierarchy of tasks.

1 Introduction

Modularity endows neural models with an inductive bias towards systematic generalisation, by activating and updating their knowledge sparsely (Hupkes et al., 2020). For instance, modules may compete to attend different parts of a structured input (Goyal et al., 2021, 2020) or disentangle pre-determined skills, reusable and autonomous

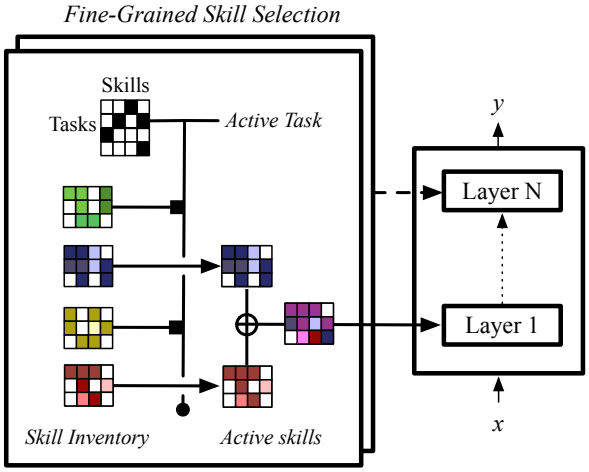

Figure 1: A diagram of our latent-skill model: 1) a row of the task–skill binary matrix is selected according to the active task; 2) the (sparse or low-rank) parameters corresponding to active skills from a layer-specific inventory are combined; 3) the resulting parameterisation is plugged into a neural network.

facets of knowledge, across multiple tasks to be later recombined in original ways for new tasks (Alet et al., 2018; Ponti, 2021; Ansell et al., 2021; Kingetsu et al., 2021). These two levels of modularity mirror two distinct levels of memory (Hill et al., 2021; Yogatama et al., 2021)—short-term for input-level knowledge and long-term for task-level knowledge—and ultimately reflect the integrated yet modular nature of the cognitive system in humans (Clune et al., 2013).

In multitask learning, previous work focused on settings where the skills relevant for each task are known *a priori* (Pfeiffer et al., 2020; Ponti et al., 2021a; Ansell et al., 2022), or settings where parameters or representations are an entangled mixture of shared and task-specific knowledge (Misra et al., 2016; Ruder et al., 2019; Karimi Mahabadi et al., 2021). The former method requires expert knowledge (possibly sub-optimal and limited to a few domains), whereas the latter leaves multitask

models vulnerable to distribution shifts and hence hinders them from quickly adapting to new tasks. To remedy these shortcomings, in this work we address a setting where the skills needed for each task are modular but unknown and possibly finer-grained than those posited by experts. Thus, we propose a method to jointly learn, in an end-to-end fashion: 1) a task–skill allocation matrix, which indicates which subset of skills (from a fixed inventory) are active for which task; and 2) a corresponding set of parameter-efficient adaptations of the model, a subset of which are superimposed to a base model according to the active skills.

The proposed model is equivalent to performing a soft partition, represented by a binary matrix, of the set of skill-specific parameters (Orbanz, 2012). Thus, an Indian Buffet Process (Griffiths and Ghahramani, 2005; Teh et al., 2007) can be posited as a prior over this matrix, to regularise it towards striking a balance in allocating different subsets of skills to each task. As an alternative inductive bias, we also explore using a higher learning rate for the task–skill matrix compared to the skill-specific parameters, as it promotes better allocations over general-purpose parameterisations.

We evaluate our model on both reinforcement and supervised learning. For the first set of experiments, we focus on BabyAI (Chevalier-Boisvert et al., 2019), a platform consisting of a sequence of levels where agents must follow linguistic instructions in a simulated environment. Each level is procedurally generated to reflect an increasingly complex combination of skills (e.g., picking up objects or unlocking doors). We find that our modular network achieves higher sample efficiency—by requiring a significantly smaller number of episodes to reach a near-perfect success rate in all levels—compared to all baselines including fully shared and level-specific models based on a state-of-the-art architecture (Hui et al., 2020), as well as a model that has access to the ground-truth skills.

In addition, in the wake of the recent surge of interest in massively multitask few-shot NLP models (Min et al., 2021; Wei et al., 2021; Aribandi et al., 2021; Sanh et al., 2022; Karimi Mahabadi et al., 2021, *inter alia*), we also evaluate our latent-skill model on CrossFit (Ye et al., 2021). This benchmark recasts 160 NLP tasks (including QA, conditional text generation, classification, and other types such as regression) as text-to-text generation problems. We obtain superior performance in few-shot adaptation to held-out tasks compared to a series of competitive baselines that include HyperFormer, a state-of-the-art method for task-conditional parameter generation (Karimi Mahabadi et al., 2021). Thus, we demonstrate the ability of our model to successfully reuse and adjust the skills, which were previously acquired during multitask pre-training, on new tasks.

Moreover, we show that our method can be used in tandem with several parameter-efficient methods (He et al., 2021) in order to make the increase in time and space complexity due to skill-specific parameters negligible. In particular, we explore sparse adaptation with Lottery-Ticket Sparse Fine-Tuning (LT-SFT; Ansell et al., 2022) and low-rank adaptation with Low-Rank Adapters (LoRA; Hu et al., 2021). Finally, in addition to sample efficiency and systematic generalisation, we illustrate how our method also favours interpretability, as it discovers explicitly which pairs of tasks require common modules of knowledge.

The code for our model is available at: `github.com/McGill-NLP/polytropon`.

2 A Latent-Skill Multitask Model

The goal of multitask learning in modelling a set of tasks $\mathcal{T} = (\mathcal{T}_1, \ldots, \mathcal{T}_{|\mathcal{T}|})$ is two-fold: 1) increasing sample efficiency on each seen task by borrowing statistical strength from the others; and 2) attaining systematic generalisation, the ability to adapt robustly to new tasks, possibly based on a few target-domain examples. In particular, in supervised learning, each task $\mathcal{T}_i$ is associated with a dataset $\mathcal{D}_i \triangleq \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ and a loss function $\mathcal{L}(\hat{y}, y)$, where each $\mathbf{x}$ is an input and each y is a label. In reinforcement learning, each task is characterised by an initial state distribution $q(\mathbf{x}_1)$, a transition distribution $q(\mathbf{x}_{t+1} \mid \mathbf{x}_t, a_t)$, and a loss function $\mathcal{L}(\mathbf{x}_1, a_1, \ldots, \mathbf{x}_h, a_h) \rightarrow \mathbb{R}$,[1] where $\mathbf{x}$ is a state, a is an action, and h is the temporal horizon of each episode. Thus, each task defines a Markov Decision Process (MDP).

Previous work sought to counter the limitations of fully sharing the parameters of a model across all tasks (Stickland and Murray, 2019), which may exhaust model capacity and lead to interference among the task-specific gradients (Wang et al., 2021). Instead, parameters can be softly shared across tasks by composing task-specific adapters (Pfeiffer et al., 2021) or generating parameters

[1] This is the negative of the reward: $\mathcal{L}(\cdot) = -\mathcal{R}(\cdot)$.

via task-conditioned hyper-networks (Ponti et al., 2021a; Karimi Mahabadi et al., 2021; Ansell et al., 2021). The first method leads to an explosion in parameter count, which grow linearly with the size of the number of tasks. Moreover, due to entangling knowledge across tasks, the second method suffers during few-shot adaptation to new tasks as it may overfit the training task distribution.

In this work, we posit instead that there exists a (possibly small) fixed inventory of skills $\mathcal{S} = (\mathcal{S}_1, \ldots, \mathcal{S}_{|\mathcal{S}|})$, where $|\mathcal{S}| \ll |\mathcal{T}|$. Each skill is an independent facet of knowledge that is reused across a subset of tasks. These assumptions guarantee both scalability and modularity. In particular, we seek to create a model that jointly learns which skills are active for which task, aggregates the corresponding skill parameters according to some deterministic function, and maximises the multitask log-likelihood

$$\sum_{\mathcal{T}_i} \sum_{(\mathbf{x},y)\in\mathcal{T}_i} \log p(y \mid \mathbf{x}, Z, \Phi, \mathcal{T}_i)\, p(Z \mid \alpha)\, p(\Phi) \tag{1}$$

with respect to two distinct sets of parameters: i) a matrix Z of soft partitions of skills across tasks, regularised by a prior with hyper-parameter α; ii) a matrix Φ of skill-specific parameters for a given neural architecture, which are composed according to the active skills. The full graphical model is shown in Figure 2. In what follows, we illustrate each component separately.

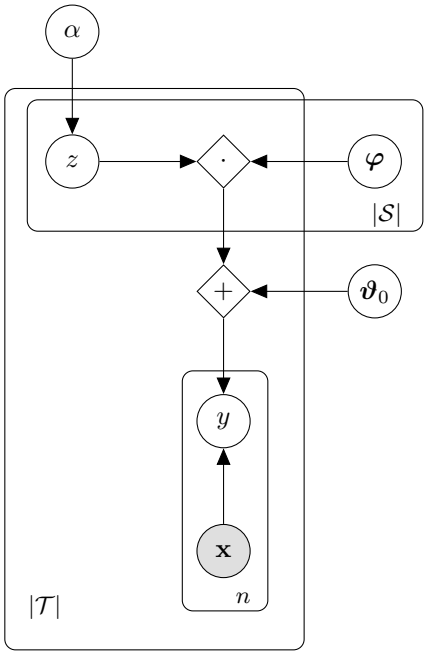

Figure 2: A graph (plate notation) of the generative model of neural modules. Shaded circles refer to observed variables.

2.1 Soft Partitions

What is the best strategy to determine which skills are active for which task? The cognitively inspired notion of modularity at the level of structured inputs assumes that modules compete with each other for activation and updating (Bengio, 2017; Goyal et al., 2021). This intuition is translated in practice into a softmax across modules and top-k selection. Instead, we argue, modularity at the task level should reflect the fact that tasks fall into a hierarchy where more complex ones subsume simpler ones (e.g., dialogue requires both intention classification and text generation). Hence, variable-size subsets of skills should be allowed.

As a consequence, we assume that the matrix $Z \in \{0,1\}^{|\mathcal{T}|\times|\mathcal{S}|}$ representing task–skill allocations is a soft partition: each cell z_{ij} is a binary scalar indicating if module φ_j is active for a certain task $\mathcal{T}_i$. However, being discrete, such a binary matrix is not differentiable and therefore cannot be learned end-to-end via gradient descent. Instead, we implement it as a collection of continuously relaxed Bernoulli distributions through a Gumbel-sigmoid (Maddison et al., 2017; Jang et al., 2017), which ensures stochasticity while allowing for differentiable sampling:

$$\hat{z}_{i,j} = \sigma\left[\log \frac{\sigma(z_{i,j})\, u}{(1-\sigma(z_{i,j}))\,(1-u)}^{1/\tau}\right]$$
$$u \sim \text{Uniform}(0,1). \tag{2}$$

In principle, either a coarse-grained soft partition can be learned globally for the entire neural network, or different fine-grained soft partitions can be assigned to each layer. We opt for the second alternative as it affords the model more flexibility and, foreshadowing, yields superior performance. Therefore, Z and Φ are henceforth assumed to be layer-specific, although we will omit layer indexes in the notation for simplicity's sake.

2.2 Skill-specific Parameters

Given the matrix row $\hat{Z}_{i,\star}$ for task $\mathcal{T}_i$ from Equation (2) and a matrix of skill-specific parameters $\Phi \in \mathbb{R}^{|\mathcal{S}|\times d}$, where d is the dimension of the layer parameters, the aggregate of the parameters of active skills is superimposed to a base parameterisation $\boldsymbol{\vartheta}_0 \in \mathbb{R}^d$ shared across tasks. For instance, $\boldsymbol{\vartheta}_0$ may be either the initialisation from a pre-trained

model or learned from scratch:

$$p(y|\mathbf{x}, Z, \Phi, \mathcal{T}_i) \triangleq p(y|x, \boldsymbol{\vartheta}_i)$$
$$\boldsymbol{\vartheta}_i = \boldsymbol{\vartheta}_0 + \sum_{\mathcal{S}_j} \Phi_{j,\star} \frac{\hat{z}_{i,j}}{\sum_{\mathcal{S}_j} \hat{z}_{i,j}}. \quad (3)$$

Note that we normalise the rows of the task–skill allocation matrix $\hat{Z}$ prior to composition because the variable number of active skills per task would otherwise affect the norm of the combined parameters $\boldsymbol{\vartheta}_i$, thus making training unstable.

2.3 Inductive Biases

A possible failure mode during training is a collapse into a highly entropic or non-sparse allocation matrix $\hat{Z}$, where all skills are active and skill-specific parameters remain general-purpose rather than specialising. Thus, we also provide an inductive bias to encourage the model to learn a low-entropy, highly-sparse allocation matrix.

IBP Prior A possible inductive bias consists of adding an Indian Buffet Process (IBP; Teh et al., 2007) prior to Z, as IBP is the natural prior for binary matrices representing soft partitions. In particular, this assumes that the i-th task is associated with skills used by previous tasks with probability $\text{Bernoulli}(m_{\mathcal{S}_j}/i)$ (i.e., in proportion to their 'popularity') and $\text{Poisson}(\alpha/i)$ new skills, where $m_{\mathcal{S}_j} = \sum_{\mathcal{T}_i} z_{i,j}$ is the count of tasks for which skill $\mathcal{S}_j$ is active. The log-probability of a task–skill allocation matrix under an IBP prior with hyper-parameter α is:

$$\log p(Z \mid \alpha) = |\mathcal{S}| \log \alpha - \sum_{h=1}^{2^{|\mathcal{T}|}-1} \log Z_h! - \alpha \sum_{\mathcal{T}_i} H_{\mathcal{T}_i} + \sum_{\mathcal{S}_j} \big[\log(|\mathcal{T}| - m_{\mathcal{S}_j})! + \log(m_{\mathcal{S}_j} - 1)! - \log(|\mathcal{T}|)!\big] \quad (4)$$

where H_n is the n-th harmonic number and Z_h is the number of skills possessing the history h (the binary vector of their corresponding column in the matrix Z). While training a neural network, the prior probability in Equation (4) can be taken into account in the form of a regulariser subtracted to the main loss function.

Two-speed Learning Rate As a simpler inductive bias alternative to the IBP prior, we also experiment with setting the learning rate for Z higher than for Φ. Intuitively, by accelerating learning of the soft partition matrix, to minimise the loss it becomes more convenient to discover better task–skill allocations over settling for general-purpose parameters that are agnostic with respect to the subset of active skills.

2.4 Parameter Efficiency

In order to keep the skills modular, each of them must correspond to a separate layer parameterisation. Nevertheless, this may lead to a significant increase in both time and space complexity. Thus, we explore parameter-efficient implementations of Φ that only add a negligible amount of parameters to the base model. In particular, we contemplate both sparse and low-rank approximations.

Sparse Approximations Lottery Ticket Sparse Fine-Tuning (LT-SFT; Ansell et al., 2022) learns a highly sparse vector of differences with respect to a base model $\boldsymbol{\vartheta}_0$. In our setting, this amounts to identifying a binary matrix $M \in \{0,1\}^{|\mathcal{S}| \times d}$ indexing non-zero entries in Φ.[2] We infer M by selecting the top-k entries in Φ based on their change in magnitude after a few early episodes:

$$\mu_{i,j} \leftarrow \begin{cases} 1 & \text{if } \phi_{i,j} \in \text{argmax}_{\phi_1,\ldots,\phi_k} |\Phi' - \Phi| \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

The advantage of LT-SFT over other methods is that it is architecture-agnostic. However, it suffers from high space complexity during the early phase of training.

Low-rank Approximations Other parameter-efficient methods are designed for Transformer architectures specifically. For instance, Low-Rank Adapters (Hu et al., 2021) factorise each weight of the linear projections inside self-attention layers as a multiplication between two low-rank matrices. Hence, a linear projection $f_{\boldsymbol{\varphi}} : \mathbb{R}^i \to \mathbb{R}^o$ is implemented as:

$$\mathbf{x}' = [W_0 + (\mathbf{z}^\top AB)]\mathbf{x} + \mathbf{b}_0 \quad (6)$$

where $A \in \mathbb{R}^{|\mathcal{S}| \times o \times r}$, $B \in \mathbb{R}^{|\mathcal{S}| \times r \times i}$, and r is the rank. Hence, $\Phi \triangleq \text{flatten}(AB)$.

2.5 Baselines

We measure the performance of our SKILLED approach, where we learn the skill–task allocation matrix Z end-to-end, against the following baselines:

[2] We leverage PyTorch `sparse_coo_tensor` for implementation.

- PRIVATE: there is a separate model parameterisation for each task. During few-shot adaptation, given that $\mathcal{T}_{train} \cap \mathcal{T}_{eval} = \emptyset$, this model cannot benefit from any transfer of information between training and evaluation tasks. This is equivalent to the special case where the task–skill allocation matrix is an identity matrix $Z = I$ of size $|\mathcal{T}| \times |\mathcal{T}|$ and $|\mathcal{S}| = |\mathcal{T}|$.
- SHARED: a shared skill is learnt on the training tasks and then fine-tuned for each evaluation task separately. This is equivalent to the special case where the task–skill allocation matrix is a matrix of ones $Z = \mathbf{1}$ of size $|\mathcal{T}| \times 1$ and $|\mathcal{S}| = 1$.
- EXPERT, where the task–skill allocation is contingent on expert knowledge about task relationships. Crucially, Z is fixed *a priori* rather than being learned.

In addition, we compare our SKILLED method to a state-of-the-art baseline for multitask learning where parameters are softly shared, HYPERFORMER (Karimi Mahabadi et al., 2021). This method takes inspiration from Ponti et al. (2021a) and generates adapters for the pre-trained model parameters with hyper-networks conditioned on task embeddings. In particular, parameters for task $\mathcal{T}_i$ are obtained as:

$$\mathbf{x}' = [W_0 + f_A(\mathcal{T}_i)\, f_B(\mathcal{T}_i)]\mathbf{x} + \mathbf{b}_0 \qquad (7)$$

where each $f_W(\mathcal{T}_i) \triangleq W_2[\text{ReLU}(W_1 \text{e}(\mathcal{T}_i))]$ is a hyper-network, $\text{e}(\mathcal{T}_i)$ is the embedding of task $\mathcal{T}_i$, $W_1 \in \mathbb{R}^{h \times e}$ and $W_2 \in \mathbb{R}^{d \times h}$ for hidden size h and task embedding size e. Note that, in our case, we generate LoRA rather than Adapter layers, contrary the original formulation of Karimi Mahabadi et al. (2021), in order to make the baseline comparable to the implementation of SKILLED in Equation (6).

3 Reinforcement Learning Experiments

3.1 Dataset

As a proof-of-concept experiment, we perform multitask reinforcement learning on the BabyAI platform (Chevalier-Boisvert et al., 2019). This benchmark consists in a series of increasingly complex levels, where an agent must execute a linguistic command by navigating a two-dimensional grid world and manipulating objects. Crucially, levels are procedurally generated to reflect different subsets of skills (e.g., PICKUP, UNLOCK, ...). This enables us to test our model in a controlled setting where performance based on learned skills can be compared with 'ground truth' skills. In particular, we focus on a similar subset of 8 levels as Hui et al. (2020): GOTOOBJ, GOTOREDBALLGREY, GOTOREDBALL, GOTOLOCAL, PUTNEXTLOCAL, PICKUPLOC, GOTOOBJMAZE, GOTO.

During each episode within a level, the visual input is a 7 by 7 grid of tiles, a partial observation of the environment based on the agent's field of view at the current time step. Each tile consists of 3 integers corresponding to the properties of a grid cell: object type, colour, and (optionally for doors) if they are open, closed, or locked. The linguistic instructions in English may require to complete multiple goals (via coordination) or express a sequence of sub-goals (via temporal markers).

3.2 Experimental Setup

Model Architecture The neural architecture adheres to the best model reported in Hui et al. (2020), BOW_ENDPOOL_RES. It encodes the linguistic input through a Gated Recurrent Unit (GRU; Cho et al., 2014) and the visual input through a convolutional network (CNN). These two streams from different modalities are then merged into a single representation through FiLM (Perez et al., 2018). This component performs a feature-wise affine transformation of the CNN output conditioned on the GRU output.

Afterwards, a Long Short-Term Memory network (LSTM; Hochreiter and Schmidhuber, 1997), a recurrent module keeping track of the agent state trajectory, receives the multimodal representation and returns the current hidden state. This in turn is fed into two distinct MLPs, an actor and a critic (Sutton, 1984). The actor yields a distribution over actions, whereas the critic a reward baseline for the current state. In our experiments, each row of the matrix Φ corresponds to a possible parameterisation for all these components.

To determine *a priori* a skill–task allocation for the EXPERT baseline, we harness the information about the skills employed to procedurally generate each level by Chevalier-Boisvert et al. (2019, p. 6), which are indicated in Table 2. For the SKILLED model, we set $|\mathcal{S}| = 9$ similarly to EXPERT. This allows us to compare learned skills and 'ground-truth' skills from an inventory of identical size. As a parameter-efficient implementation of Φ, we em-

ploy LT-SFT (Ansell et al., 2022) with a sparsity of 90%. For all model variants, $\boldsymbol{\vartheta}_0$ and Φ are both initialised from a Kaiming uniform and learnable. During training, we sample levels uniformly.

Hyper-parameters We follow closely the best hyper-parameter setup of Hui et al. (2020). Tiles in the visual input are encoded into embeddings of size 128 via a look-up table. The CNN has 2 layers, filter size 3, stride 1, and padding 1; whereas the FiLM module has 2 layers. A residual layer is added between the CNN output and each of the FiLM layers. The output of FiLM is max-pooled with a layer of size 7 and stride 2. Both the LSTM and the GRU have a hidden size of 128.

A learning rate of 1e-4 is adopted for Adam (Kingma and Ba, 2015). We optimise the model with Proximal Policy Optimisation (PPO; Schulman et al., 2017) and Back-Propagation Through Time (BPTT; Werbos, 1990). Additionally, we use an Advantage Actor–Critic (A2C; Wu et al., 2017) with Generalised Advantage Estimation (GAE; Schulman et al., 2015). The reward is calculated as $(1-0.9\,n/n_{\max})$ if the agent completes a task—where n is the number of steps required and $n_{\max}$ is a threshold set according to the level difficulty, 0 otherwise. Returns are discounted by $\gamma = 0.99$.

3.3 Results

We now measure whether our latent-skill model facilitates sample efficiency, which following Chevalier-Boisvert et al. (2019) is defined as the number of episodes required for an agent to reach a success rate greater than 0.99. Success in turn is defined as executing an instruction in a number of steps $n < n_{\max}$, where the threshold again depends on the level complexity.

We plot our results in Figure 3. Firstly, models sharing information across tasks (either fully or mediated by skills) enjoy higher sample efficiency than assigning disjoint parameters for each task (PRIVATE), as they can borrow statistical strength from each other. Crucially, among information-sharing models, SKILLED (where knowledge is modular) surpasses SHARED (where knowledge is entangled among tasks). Thus, considering a task as a collection of fine-grained skills that can be separated and reused is the most effective way of sharing information. Finally, results surprisingly show that learning a task–skill allocation matrix end-to-end (SKILLED) is more beneficial than leveraging the ground-truth task–skill

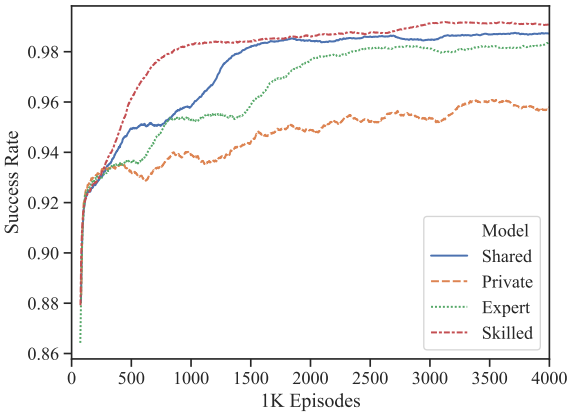

Figure 3: Smoothed sample efficiency (success rate vs. number of episodes) for different multitask models across 8 levels of BabyAI.

decomposition used to create the BabyAI levels (EXPERT). This highlights the fact that different tasks might mutually benefit in ways that go beyond what is posited *a priori* by experts, and that our proposed approach can successfully uncover and exploit such task synergies.

Moreover, we run several ablations to study the impact of the inductive biases and the parameter-efficient implementation. We report the number of episodes required to reach a success rate > 0.99 in Table 3, comparing the SKILLED model in the standard setup (with two-speed learning rates and sparse skill-specific parameters) with other variants (with an IBP prior or fully dense skill-specific parameters). We find that adding an IBP prior with $\alpha = 5$ does not affect the performance significantly. Hence, for the remainder of the experiments, we will adopt two-speed learning rates as an inductive bias. On the other hand, employing fully dense skill-specific parameters increases sample efficiency, albeit to a limited degree. Thus, we verify that parameter sparsification is an effective trade-off between performance and space complexity.

4 Supervised Learning Experiments

4.1 Dataset

In order to measure the benefits of a modular design for systematic generalisation to new tasks, we run a second set of experiments on CrossFit (Ye et al., 2021), a benchmark including 160 diverse natural language processing tasks sourced from Huggingface Datasets (Lhoest et al., 2021). The tasks in CrossFit are all converted into a unified

TASK	Metric	Ye et al. (2021)	Shared	Private	Expert	HyperFormer	Skilled
AG-NEWS	C–F1	84.6 ± 1.4	59.6 ± 21.1	74.7 ± 10.2	47.0 ± 9.9	64.6 ± 13.4	81.2 ± 8.0
AI2-ARC	Acc	22.8 ± 1.9	23.7 ± 2.4	20.3 ± 0.8	28.3 ± 3.7	23.8 ± 6.7	22.3 ± 3.4
AMAZON-POLARITY	C–F1	92.2 ± 0.6	92.4 ± 2.8	94.4 ± 0.4	57.4 ± 3.1	93.8 ± 1.5	93.7 ± 0.9
BSN-NPI-LICENSOR	Acc	99.9 ± 0.2	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.0	99.9 ± 0.2
BSN-NPI-SCOPE	Acc	85.7 ± 13.0	99.8 ± 0.3	99.6 ± 0.9	99.9 ± 0.2	99.9 ± 0.0	99.9 ± 0.2
BREAK-QDMR	EM	4.8 ± 0.4	4.1 ± 0.5	1.9 ± 1.7	4.1 ± 0.9	3.8 ± 1.2	4.9 ± 0.6
CIRCA	C–F1	42.3 ± 7.8	44.3 ± 7.5	22.2 ± 6.3	13.0 ± 7.0	25.0 ± 6.3	45.9 ± 5.7
CRAWL-DOMAIN	EM	20.7 ± 2.0	40.9 ± 2.2	36.2 ± 6.3	37.1 ± 5.3	34.9 ± 3.8	39.0 ± 4.2
ETHOS-DISABILITY	C–F1	75.3 ± 2.2	66.9 ± 11.8	64.7 ± 12.1	59.2 ± 15.8	79.4 ± 3.9	72.2 ± 5.2
ETHOS-SEXUAL	C–F1	59.7 ± 5.7	62.4 ± 8.4	71.2 ± 9.8	40.3 ± 11.4	76.8 ± 10.0	86.1 ± 2.6
FREEBASE-QA	EM	1.3 ± 0.1	0.7 ± 0.2	0.2 ± 0.1	1.3 ± 0.5	1.6 ± 0.8	0.7 ± 0.3
GLUE-COLA	M-Corr	3.5 ± 6.7	12.9 ± 5.5	9.1 ± 6.3	7.6 ± 5.6	6.8 ± 4.2	7.1 ± 5.3
GLUE-QNLI	Acc	74.7 ± 2.9	75.5 ± 3.6	57.1 ± 7.7	56.6 ± 19.7	73.9 ± 3.2	78.1 ± 1.6
HATEXPLAIN	C–F1	44.9 ± 2.5	33.1 ± 8.2	26.5 ± 7.8	11.9 ± 4.0	23.2 ± 6.2	32.6 ± 13.6
QUOREF	QA-F1	41.2 ± 1.6	46.0 ± 4.4	36.3 ± 4.6	48.4 ± 4.3	41.7 ± 6.5	47.3 ± 3.5
RACE-HIGH	Acc	30.5 ± 1.5	34.0 ± 2.7	28.5 ± 1.4	38.5 ± 2.0	30.8 ± 1.9	34.8 ± 2.0
SUPERGLUE-RTE	Acc	60.4 ± 3.6	60.6 ± 2.9	49.7 ± 5.1	51.7 ± 4.8	60.9 ± 3.8	60.4 ± 5.9
TWEET-EVAL-IRONY	C–F1	55.2 ± 3.6	52.1 ± 8.0	50.1 ± 14.2	25.6 ± 8.9	38.4 ± 6.0	57.2 ± 2.4
WIKI-SPLIT	Rouge-L	79.3 ± 0.5	80.1 ± 0.6	80.3 ± 0.6	79.2 ± 0.8	79.2 ± 0.7	80.6 ± 0.3
YELP-POLARITY	C–F1	71.8 ± 21.1	88.3 ± 14.9	65.0 ± 20.5	53.9 ± 12.7	95.0 ± 0.9	94.5 ± 1.1
ALL	Avg	52.5 ± 30.8	53.9 ± 30.5	49.4 ± 31.7	43.0 ± 29.0	52.7 ± 33.2	**56.9 ± 32.3**

Table 1: Few-shot adaptation results of SKILLED and four baselines, as well as the original model from Ye et al. (2021). Results are reported both in aggregate and separately for each task in $\mathcal{T}_{eval}$. For the full name of the metrics, refer to Section 4.1.

text-to-text format inspired by Raffel et al. (2020). Moreover, they are partitioned into three disjoint subsets. First, a model is pre-trained in a multi-task fashion on training tasks $\mathcal{T}_{train}$. Afterwards, it is adapted to each evaluation task from $\mathcal{T}_{eval}$ in a few-shot learning setting. Hyper-parameter are tuned on the held-out set $\mathcal{T}_{dev}$.

We adopt the partition 1 (called RANDOM) from Ye et al. (2021), where $|\mathcal{T}_{train}| = 120$, $|\mathcal{T}_{dev}| = |\mathcal{T}_{eval}| = 20$, and tasks are split randomly. This is the most comprehensive partition and most suited for general-purpose models, as it includes all types of tasks. Every task is associated with 5 different few-shot data splits for train $\mathcal{D}_{train}$ and development $\mathcal{D}_{dev}$ and 1 larger data split for evaluation $\mathcal{D}_{eval}$. During multitask pre-training, we concatenate all train and development splits of datasets from $\mathcal{T}_{train}$, whereas during few-shot adaptation we use the splits of datasets in $\mathcal{T}_{eval}$ separately in 5 distinct runs. We measure the performance of a model with 7 evaluation metrics according to the type of task: C[lassification]-F1, Acc[uracy], QA-F1, E[xact] M[atch], Rouge-L, M[atthew]-Corr[elation], and P[earson]-Corr[elation].

4.2 Experimental Setup

Model Architecture As pre-trained weights ϑ_0 for conditional text generation, we choose BART Large (Lewis et al., 2020), a 24-layer Transformer-based encoder–decoder. We use LoRA (Hu et al., 2021) to implement skill-specific parameters efficiently, as it was explicitly designed for the Transformer architecture and has lower space complexity that LT-SFT during the early phases of training (cf. Section 2.4). While ϑ_0 remains frozen for all model variants, A matrices in Φ (and $f_A(\cdot)$ in HYPERFORMER) are initialised to zero matrices following Hu et al. (2021).

As a source of expert knowledge for the EXPERT baseline, we associate each task to a unique skill corresponding to one of the 4 task types of Ye et al. (2021, p. 20)'s taxonomy: question answering, conditional text generation, classification, or other (e.g., regression). The skill inventory size for SKILLED was chosen among $\{2, 4, 8, 16, 32\}$ based on validation. We set the embedding size e and hidden size h of HYPERFORMER to an identical value to ensure a fair comparison.

Hyper-parameters During both multitask pre-training and few-shot adaptation, we use the Adam optimiser (Kingma and Ba, 2015) and select a learning rate for Φ among $\{1e-2, 1e-3, 1e-4\}$ based on performance on the development set of $\mathcal{T}_{train}$ and $\mathcal{T}_{dev}$, respectively. As a more aggressive learning rate for Z in SKILLED instead we search in the range $[1e-1, 1e-2]$. We run mul-

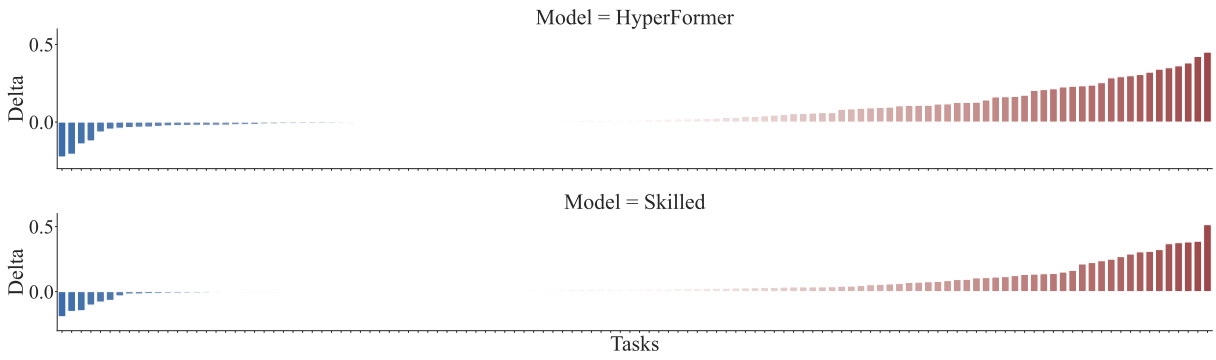

Figure 4: Delta in performance in terms of task-specific metrics between SKILLED (top) and HYPERFORMER (bottom) on the one hand and SHARED on the other, across 120 seen CrossFit tasks after multitask pre-training.

titask pre-training for 30 epochs with an effective batch size of 32, with a warm-up of 6% of the total steps. Instead, during few-shot adaptation, the effective batch size is 8 for 1000 training steps, with a warm-up rate of 10% and a weight decay of 1e-2.

For a comparison of parameter counts, LoRA adds $4l(2hr + |\mathcal{T}|) \cdot |\mathcal{S}|$ parameters to the pre-trained model, where l is the number of layers in the encoder and decoder, h is the hidden size, and 4 is the number of linear projections in self-attention (query, key, value, output). We use $r = 16$, $l = 24$ and $h = 1024$ for BART Large, so we add $\sim 3 \cdot 10^6$ parameters per skill. Given that the pre-trained model has $\sim 4 \cdot 10^8$ parameters, this implies an increase of $\sim 0.78\%$ per skill. HYPERFORMER adds $4l(2he + 2e) \cdot e$ parameters, an increase of $\sim 0.78\%$ per task embedding dimension.

4.3 Results

Few-shot Adaptation to New Tasks In the few-shot adaptation setting, our goal is to evaluate the capability of the model to quickly generalise to new tasks unseen during training. This is the most realistic setting as tasks encountered by models deployed 'in the wild' will be characterised by different distributions or involve different input / output spaces. Performance in terms of task-specific metrics is reported in Table 1 for the 20 evaluation tasks individually and on average.

Crucially, results show that SKILLED outperforms alternative formulations of the task–skill allocation matrix, such as SHARED, PRIVATE and EXPERT. Importantly, we note that SKILLED also surpasses HYPERFORMER by a sizeable margin despite the two models having comparable parameter counts. This points to the fact that explicitly modularising knowledge learnt during multitask training is important for systematic adaptation to unseen tasks, whereas entangled knowledge is more brittle to distribution shifts in cross-task transfer. Finally, we corroborate the soundness of our experimental setup by reproducing the results of Ye et al. (2021).[3]

Multitask Evaluation on Seen Tasks Moreover, to ensure that modularity does not adversely affect in-domain performance, we evaluate models on the test sets of seen tasks after multitask training. The results are shown in Table 4. SKILLED achieves average improvements of up to ~3% in the global metrics with respect to SHARED by flexibly allocating skills to each task. In Figure 4, we report the delta in performance in terms of task-specific metrics between SKILLED and SHARED for the 120 training tasks, in most of which SKILLED yields positive gains. In this context, we can see that SKILLED achieves comparable performance to HYPERFORMER, therefore confirming that explicit modularisation can be as effective as conditional parameter generation when evaluated on seen tasks, but also engenders vast improvements on held-out tasks.

In-Depth Analysis of Learned Skills Finally, we run an in-depth analysis of the task–skill allocation matrices Z learned by SKILLED. Specifically, we measure:

1. *Discreteness*. How close is the continuous relaxation to a binary matrix? To this end, we report the average normalised entropy across all probabilities in the cells of the matrix:

$$\text{Discrete}(Z) = \frac{1}{|\mathcal{T}| \cdot |\mathcal{S}|} \sum_{\mathcal{T}_i} \sum_{\mathcal{S}_j} \frac{\mathcal{H}(z_{ij})}{\log 2} \quad (8)$$

[3]Note that in Ye et al. (2021) the pre-trained model is BART Small and all parameters are fine-tuned. Hence, these results are not directly comparable.

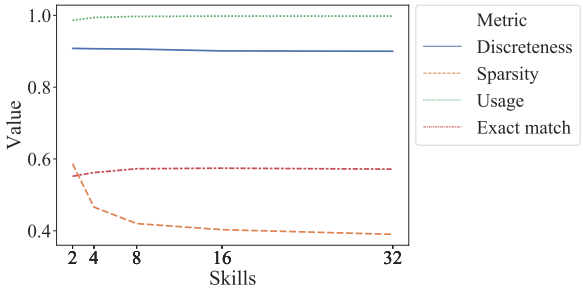

Figure 5: Statistics of the task–skill matrices for different choices of skill inventory size, including: discreteness, sparsity, usage, and the average exact match on the development set of 120 CrossFit tasks.

2. *Sparsity*. How many skills are active per task on average? We count the rate of non-zero cells in the values rounded to the closest integer:

$$\text{Sparsity}(Z) = \frac{1}{|\mathcal{T}| \cdot |\mathcal{S}|} \sum_{\mathcal{T}_i} \sum_{\mathcal{S}_j} \lfloor z_{ij} \rceil \quad (9)$$

3. *Usage*. Is the allocation of skills across tasks balanced or are some preferred over others? We provide the normalised entropy of a categorical distribution parameterised by $\sum_j Z_{\star,j}$, the sum of the columns of Z:

$$\text{Usage}(Z) = \frac{\mathcal{H}\left[\sum_{\mathcal{T}_i} z_{i,\star}\right]}{\log |\mathcal{S}|} \quad (10)$$

Note that the entropy values are normalised into the range $[0, 1]$ to make them invariant to the number of skills: this quantity is known as 'efficiency'.

We plot these metrics—as well as the performance on in-domain train tasks in terms of exact match—as a function of the skill inventory size in Figure 5. We find that, whilst a continuous relaxation, the learned matrices are highly discretised and all their values are extremely close to either 0 or 1. Moreover, the level of sparsity decreases as the number of skills increases. This means that smaller subsets of skills are required in proportion due to the diversity of available skills. Finally, usage is consistently near the maximum value, which implies that there is uniformity in how frequently each skill is active across tasks.

Overall, these results demonstrate that a quasi-binary, highly-sparse, and non-trivial allocation matrix can be successfully learned in an end-to-end fashion even with simple inductive biases such as a two-speed learning rate. For instance, we visualise the posterior of Z for $|\mathcal{S}| = 4$ in Figure 6.

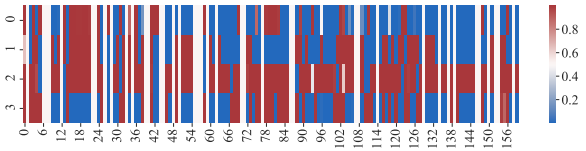

Figure 6: Posterior over Z in SKILLED for $|\mathcal{S}| = 4$.

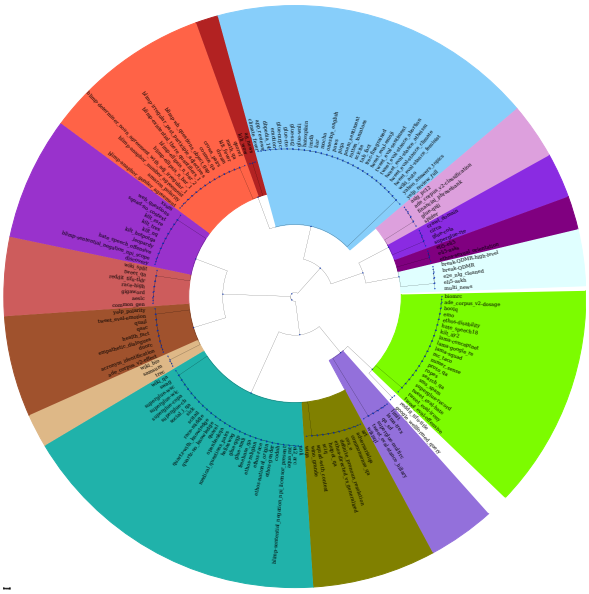

Figure 7: Task partitions for $|\mathcal{S}| = 4$, which corresponds to $2^{|\mathcal{S}|} = 16$ possible subsets of skills.

Crucially, the learned allocation also facilitates the interpretability of black-box multitask models. In fact, the structure of Z corresponds to an explicit hierarchy of tasks, where simpler ones are subsumed by more complex ones, and similar tasks can be grouped into the same category if they share the same subset of skills. We plot this hierarchy as a dendrogram in Figure 7. For instance, most GLUE tasks (Wang et al., 2018) are grouped together as they are all focused on natural language understanding: for instance, they require skill 1 (COLA), 2 (MRPC, RTE, SST2, WNLI), or both 1 and 2 (MNLI, QQP).

5 Related Work

Modular Networks The idea of modularising neural network computation by decomposing it into a subset of specialised sub-systems has long been sought as a way to achieve better generalisation to unseen inputs (Jacobs et al., 1991b; Andreas et al., 2016; Kirsch et al., 2018), tasks (Jacobs et al., 1991a; Alet et al., 2018; Ruder et al., 2019) and recently to improve continual learning (Ostapenko et al., 2021) and robustness to

changes in the environment (Goyal et al., 2021). Modular networks fall in the category of conditional computation methods, where modules are chosen dynamically given the input (Gulcehre et al., 2016). In routing networks (Rosenbaum et al., 2019), the system makes hard decisions about which modules to use and learns the structure in which modules are composed (Andreas et al., 2016; Alet et al., 2018). Andreas et al. (2016) learn the structure using an external parser while Alet et al. (2018) recur to a stochastic process in which structures are sampled with simulated annealing. In mixture of experts (MoE) approaches, the system selects a potentially sparse, soft subset of modules depending on the input to be processed (Jacobs et al., 1991b; Shazeer et al., 2017). MoEs can be interpreted from the point of view of independent mechanisms (Parascandolo et al., 2018) that Goyal et al. (2021) further extend to handle sequential problems. In the context of NLP, Fedus et al. (2021) successfully used a MoE architecture to scale large language model pre-training to trillions of parameters.

In contrast to previous approaches, our model conditions the computation on the task rather than on task inputs. There have been related attempts to enforce parameter reuse and modularity for multitask learning (Rajendran et al., 2017; Ponti et al., 2021a; Kingetsu et al., 2021; Kudugunta et al., 2021). Rajendran et al. (2017) learn separate modules for each task and then learn how to reuse those modules for a new task. Kudugunta et al. (2021) uses a set of modules for each task in a multilingual translation setting. Our approach does not assume a set of modules for each task but instead decomposes a task into a set of skills themselves reusable across tasks.

Multitask NLP Multitask learning for NLP has been an effective strategy for improving model performance in low-resource tasks and for quickly adapting to new, unseen tasks (Ruder et al., 2019; Liu et al., 2019; Min et al., 2021; Wei et al., 2021; Aribandi et al., 2021; Sanh et al., 2022; Karimi Mahabadi et al., 2021; Rusu et al., 2019), languages (Ponti et al., 2019), and modalities (Bugliarello et al., 2022). Liu et al. (2019) adopt a multitask training strategy with a shared model and achieve impressive performance on GLUE. However, the method still requires task-specific fine-tuning. Rather than re-training all the model parameters, Houlsby et al. (2019) proposes to train task-specific adapters. Pfeiffer et al. (2021) share information across task-specific adapters while alleviating negative task interference. Instead of using adapters, in our experiments we parameterise our skills with LT-SFT (Ansell et al., 2022) or LoRA (Hu et al., 2021), which achieve comparable or superior performance. Recently, Karimi Mahabadi et al. (2021) ensure cross-task information sharing by using a hypernetwork to generate task-specific adapters. Differently, our task-specific parameters are composed of a set of skills from a shared inventory, which makes our approach modular and more scalable.

Few-shot Task Adaptation Several multitask approaches specifically target adaptation to new tasks, such as meta-learning approaches (Alet et al., 2018; Rusu et al., 2019; Ponti et al., 2021b; Garcia et al., 2021; Ostapenko et al., 2021). In our paper, we efficiently achieve few-shot task adaptation by inferring the task–skill allocation matrix for new tasks and fine-tuning skill parameters, which were previously learned via multitask learning. In fact, Ye et al. (2021) found that this pre-training routine is superior to meta-learning in CrossFit. A similar attempt to recompose modular knowledge learnt on previous tasks has been recently explored by Ostapenko et al. (2021).

6 Conclusions

In this work, we argued that a modular design is crucial to ensure that neural networks can learn from a few examples and generalise robustly across tasks by recombining autonomous facets of knowledge. To this end, we proposed a model where a subset of latent, discrete skills from a fixed inventory is allocated to each task in an end-to-end fashion. The task-specific instantiation of a neural network is then obtained by combining efficient parameterisations of the active skills, such as sparse or low-rank adapters. We evaluate the sample efficiency of our model on multitask instruction following through reinforcement learning and its few-shot adaptability on multitask text-to-text generation through supervised learning. In both experiments, we surpass competitive baselines where parameters are fully shared, task-specific, combined according to expert knowledge, or generated conditionally on the task. Finally, we show that our model facilitates interpretability by learning an explicit hierarchy of tasks based on the skills they require.

References

Ferran Alet, Tomas Lozano-Perez, and Leslie P. Kaelbling. 2018. Modular meta-learning. In *Proceedings of The 2nd Conference on Robot Learning*, pages 856–868.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 39–48.

Alan Ansell, Edoardo Maria Ponti, Anna Korhonen, and Ivan Vulić. 2022. Composable sparse fine-tuning for cross-lingual transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*.

Alan Ansell, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2021. ExT5: Towards extreme multi-task scaling for transfer learning. *arXiv preprint arXiv:2111.10952*.

Yoshua Bengio. 2017. The consciousness prior. *arXiv preprint arXiv:1709.08568*.

Emanuele Bugliarello, Fangyu Liu, Jonas Pfeiffer, Siva Reddy, Desmond Elliott, Edoardo Maria Ponti, and Ivan Vulić. 2022. IGLUE: A benchmark for transfer learning across modalities, tasks, and languages. *arXiv preprint arXiv:2201.11732*.

Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. 2019. BabyAI: First steps towards grounded language learning with a human in the loop. In *International Conference on Learning Representations*.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

Jeff Clune, Jean-Baptiste Mouret, and Hod Lipson. 2013. The evolutionary origins of modularity. *Proceedings of the Royal Society B: Biological sciences*, 280(1755).

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch Transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*.

Jezabel R Garcia, Federica Freddi, Feng-Ting Liao, Jamie McGowan, Tim Nieradzik, Da-shan Shiu, Ye Tian, and Alberto Bernacchia. 2021. Meta-learning with MAML on trees. *arXiv preprint arXiv:2103.04691*.

Anirudh Goyal, Alex Lamb, Phanideep Gampa, Philippe Beaudoin, Sergey Levine, Charles Blundell, Yoshua Bengio, and Michael Mozer. 2020. Object files and schemata: Factorizing declarative and procedural knowledge in dynamical systems. *arXiv preprint arXiv:2006.16225*.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. 2021. Recurrent independent mechanisms. In *International Conference on Learning Representations*.

Thomas L. Griffiths and Zoubin Ghahramani. 2005. Infinite latent feature models and the indian buffet process. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, pages 475–482.

Caglar Gulcehre, Sarath Chandar, Kyunghyun Cho, and Yoshua Bengio. 2016. Dynamic Neural Turing Machine with soft and hard addressing schemes. *arXiv preprint arXiv:1607.00036*.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2021. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*.

Felix Hill, Olivier Tieleman, Tamara von Glehn, Nathaniel Wong, Hamza Merzic, and Stephen

Clark. 2021. Grounded language learning fast and slow. In *International Conference on Learning Representations*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *International Conference on Machine Learning*, pages 2790–2799.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

David Yu-Tung Hui, Maxime Chevalier-Boisvert, Dzmitry Bahdanau, and Yoshua Bengio. 2020. BabyAI 1.1. *arXiv preprint arXiv:2007.12770*.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795.

Robert A Jacobs, Michael I Jordan, and Andrew G Barto. 1991a. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive science*, 15(2):219–250.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991b. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for Transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 565–576.

Hiroaki Kingetsu, Kenichi Kobayashi, and Taiji Suzuki. 2021. Neural network module decomposition and recomposition. *arXiv preprint arXiv:2112.13208*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Louis Kirsch, Julius Kunze, and David Barber. 2018. Modular networks: Learning to decompose neural computation. *Advances in Neural Information Processing Systems*, 31.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3577–3599.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding.

In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The concrete distribution: A continuous relaxation of discrete random variables. In *Proceedings of the International Conference on Learning Representations 2017*.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2021. MetaICL: Learning to learn in context. *arXiv preprint arXiv:2110.15943*.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003.

Peter Orbanz. 2012. Lecture notes on Bayesian nonparametrics. Technical report, Columbia University.

Oleksiy Ostapenko, Pau Rodriguez, Massimo Caccia, and Laurent Charlin. 2021. Continual learning via local module composition. *Advances in Neural Information Processing Systems*, 34.

Giambattista Parascandolo, Niki Kilbertus, Mateo Rojas-Carulla, and Bernhard Schölkopf. 2018. Learning independent causal mechanisms. In *International Conference on Machine Learning*, pages 4036–4044.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 487–503.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An Adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673.

Edoardo Ponti. 2021. *Inductive Bias and Modular Design for Sample-Efficient Neural Language Learning*. Ph.D. thesis, University of Cambridge.

Edoardo M Ponti, Ivan Vulić, Ryan Cotterell, Marinela Parovic, Roi Reichart, and Anna Korhonen. 2021a. Parameter space factorization for zero-shot learning across tasks and languages. *Transactions of the Association for Computational Linguistics*.

Edoardo Maria Ponti, Rahul Aralikatte, Disha Shrivastava, Siva Reddy, and Anders Søgaard. 2021b. Minimax and Neyman–Pearson meta-learning for outlier languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1245–1260.

Edoardo Maria Ponti, Helen O'Horan, Yevgeni Berzak, Ivan Vulić, Roi Reichart, Thierry Poibeau, Ekaterina Shutova, and Anna Korhonen. 2019. Modeling language variation and universals: A survey on typological linguistics for natural language processing. *Computational Linguistics*, 45(3):559–601.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Janarthanan Rajendran, P Prasanna, Balaraman Ravindran, and Mitesh M Khapra. 2017. Attend, adapt and transfer: Attentive deep architecture for adaptive transfer from multiple sources in the same domain. In *International Conference on Learning Representations*.

Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. 2019. Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2019. Latent multi-task architecture learning. In *Proceedings of*

the AAAI Conference on Artificial Intelligence, pages 4822–4829.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. 2019. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M. Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*.

Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5986–5995.

Richard Stuart Sutton. 1984. *Temporal credit assignment in reinforcement learning*. Ph.D. thesis, University of Massachusetts Amherst.

Yee Whye Teh, Dilan Görür, and Zoubin Ghahramani. 2007. Stick-breaking construction for the indian buffet process. In *11th International Conference on Artificial Intelligence and Statistics (AISTATS 2007)*, pages 556–563.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2021. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Paul J Werbos. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560.

Yuhuai Wu, Elman Mansimov, Roger B Grosse, Shun Liao, and Jimmy Ba. 2017. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *Advances in neural information processing systems*, 30.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021. CrossFit: A few-shot learning challenge for cross-task generalization in NLP. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

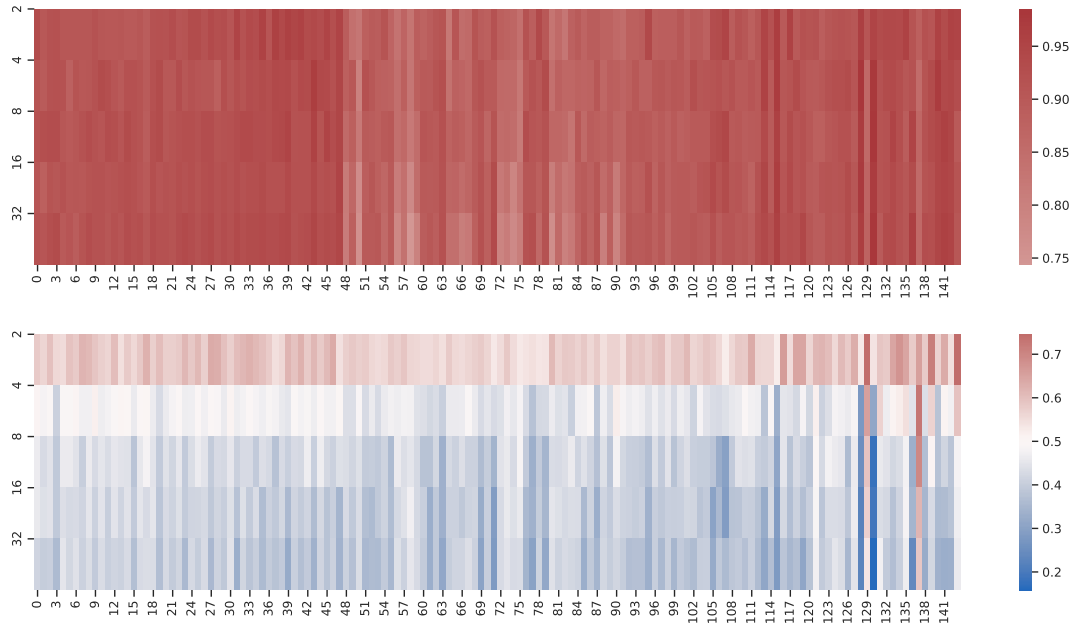

Figure 8: Per-layer discreteness (top) and per-layer sparsity (bottom).

A Additional Results for BabyAI

Skills	Level
1 2 3 4 8	GoTo
1 8	GoToObjMaze
1 2 3 6 7	PickupLoc
1 2 3 5	PutNextLocal
1 2 3 4	GoToLocal
1 2 3	GoToRedBall
1 2	GoToRedBallGrey
1	GoToObj

Table 2: BabiAI Expert task–skill allocation.

Model	Episodes
Private	>6000000
Shared	3544294
Expert	4608019
Skilled	**2218226**
+ IBP prior	2143491
- sparsity	1853060

Table 3: Sample efficiency of various models on 8 BabyAI levels measured as the number of episodes needed to reach a success rate > 0.99.

B Additional Results for CrossFit

Metric	Shared	Hyper-Former	Skilled
Task-specific			
Acc	58.47	62.63	62.66
C-F1	41.76	56.74	55.39
EM	20.05	21.68	21.70
P-Corr	56.83	61.54	52.60
QA-F1	48,88	51.04	52.35
Rouge-L	28.02	26.71	28.05
Global			
Average	43.09	49.37	48.95

Table 4: Performance of multitask models averaged over test sets of 120 seen CrossFit task. Performance is both aggregated globally across all tasks (in terms of task-specific metrics) and across subsets of tasks with the same evaluation metric.

	ELI5-ASKS, ELI5-ELI5, ETHOS-SEXUAL-ORIENTATION
3	GOOGLE-WELLFORMED-QUERY, REDDIT-TIFU-TITLE
2	APP-REVIEWS, CLIMATE-FEVER, DBPEDIA-14, EMOTION, GLUE-MRPC, GLUE-RTE, GLUE-SST2, GLUE-WNLI, HATEXPLAIN, IMDB, LIAR, MOCHA, ONESTOP-ENGLISH, PAWS, PIQA, POEM-SENTIMENT, ROTTEN-TOMATOES, SCICITE, TAB-FACT, TREC-FINEGRAINED, TWEET-EVAL-EMOJI, TWEET-EVAL-SENTIMENT, TWEET-EVAL-STANCE-ABORTION, TWEET-EVAL-STANCE-ATHEISM, TWEET-EVAL-STANCE-CLIMATE, TWEET-EVAL-STANCE-FEMINIST, WIKI-AUTO, YAHOO-ANSWERS-TOPICS, YELP-REVIEW-FULL
2 3	ADE-CORPUS-V2-DOSAGE, BIOMRC, BOOLQ, EMO, ETHOS-DISABILITY, HATE-SPEECH18, KILT-AY2, LAMA-CONCEPTNET, LAMA-GOOGLE-RE, LAMA-SQUAD, MC-TACO, NUMER-SENSE, PROTO-QA, ROPES, SEARCH-QA, SMS-SPAM, SUPERGLUE-RECORD, TWEET-EVAL-HATE, TWEET-EVAL-IRONY, TWEET-EVAL-OFFENSIVE
1	CIRCA, CRAWL-DOMAIN, GLUE-COLA, SUPERGLUE-RTE
1 3	LAMA-TREX, LIMIT, QA-SRL, SUPERGLUE-MULTIRC, TWEET-EVAL-STANCE-HILLARY, WIKISQL
1 2	AI2-ARC, ANLI, AQUA-RAT, BLIMP-SENTENTIAL-NEGATION-NPI-LICENSOR-PRESENT, CODAH, ETHOS-GENDER, ETHOS-NATIONAL-ORIGIN, ETHOS-RACE, ETHOS-RELIGION, FREEBASE-QA, GLUE-MNLI, GLUE-QQP, HELLASWAG, MEDICAL-QUESTIONS-PAIRS, OPENBOOKQA, QUAREL, QUARTZ-NO-KNOWLEDGE, QUARTZ-WITH-KNOWLEDGE, RACE-MIDDLE, SCITAIL, SICK, SOCIAL-I-QA, SUPERGLUE-CB, SUPERGLUE-COPA, SUPERGLUE-WIC, SUPERGLUE-WSC, SWAG, WIKI-QA
1 2 3	ADVERSARIALQA, ART, COMMONSENSE-QA, COS-E, DEFINITE-PRONOUN-RESOLUTION, ETHOS-DIRECTED-VS-GENERALIZED, HOTPOT-QA, SCIQ, SQUAD-WITH-CONTEXT, WINO-GRANDE, WIQA
0	BREAK-QDMR, BREAK-QDMR-HIGH-LEVEL, E2E-NLG-CLEANED, ELI5-ASKH, MULTI-NEWS
0 3	AESLC, COMMON-GEN, GIGAWORD, RACE-HIGH, REDDIT-TIFU-TLDR, TWEET-QA, WIKI-SPLIT
0 2	AG-NEWS, KILT-WOW
0 2 3	BLIMP-SENTENTIAL-NEGATION-NPI-SCOPE, DISCOVERY, HATE-SPEECH-OFFENSIVE, JEOPARDY, KILT-HOTPOTQA, KILT-NQ, KILT-TREX, KILT-ZSRE, SQUAD-NO-CONTEXT, WEB-QUESTIONS, XSUM
0 1	ADE-CORPUS-V2-CLASSIFICATION, ASLG-PC12, FINANCIAL-PHRASEBANK, GLUE-QNLI, SPIDER
0 1 3	SAMSUM, TREC, WIKI-BIO
0 1 2	AMAZON-POLARITY, BLIMP-ANAPHOR-GENDER-AGREEMENT, BLIMP-ANAPHOR-NUMBER-AGREEMENT, BLIMP-DETERMINER-NOUN-AGREEMENT-WITH-ADJ-IRREGULAR-1, BLIMP-ELLIPSIS-N-BAR-1, BLIMP-ELLIPSIS-N-BAR-2, BLIMP-EXISTENTIAL-THERE-QUANTIFIERS-1, BLIMP-IRREGULAR-PAST-PARTICIPLE-ADJECTIVES, BLIMP-WH-QUESTIONS-OBJECT-GAP, COSMOS-QA, CROWS-PAIRS, DREAM, KILT-FEVER, MATH-QA, QUOREF
0 1 2 3	ACRONYM-IDENTIFICATION, ADE-CORPUS-V2-EFFECT, DUORC, EMPATHETIC-DIALOGUES, HEALTH-FACT, QASC, QUAIL, TWEET-EVAL-EMOTION, YELP-POLARITY

Table 5: Skill allocation to 120 training CrossFit tasks for $|\mathcal{S}| = 4$.