A Extended related work

Multi-task learning. [44] demonstrated that negative transfer can worsen generalization performance, and avoidance of negative transfer has motivated much work on hierarchical Bayes in transfer learning and domain adaptation [*e.g.,* 29, 57, 16, 11, 54]. Closest to our proposed approach is early work on hierarchical Bayesian multi-task learning with neural networks that places a prior only on the output layer [24, 2, 46, 50]. In contrast, we place a non-parametric prior on the full set of neural network weights. Furthermore, none of these approaches were applied to the episodic training setting of meta-learning. Similar to our point estimation procedure, [24] and [50] propose training a mixture model over the output layer weights of a neural network using MAP inference. However, these approaches do not scale well to all the layers in a network as performing full passes on the dataset for inference of the full set of weights is computationally intractable in general.

Clustering. Incremental or stochastic clustering was considered in the EM setting in [37]. and in the K-means setting in [48]. [31] conducted online learning of a non-parametric mixture model using sequential variational inference. A key distinction between our work and these approaches is that we leverage the connection between empirical Bayes in a hierarchical model and gradient-based meta-learning [21] to use a MAML-like [14] objective as a log posterior surrogate. This allows our algorithm to make use of a scalable stochastic gradient descent optimizer instead of alternating a maximization step with an inference pass over the full dataset [*c.f.,* 50, 3].

Our approach is also distinct from recent work on gradient-based clustering [22] since we employ the episodic batching of [53]. This can be a challenging setting for a clustering algorithm, as the assignments need to be computed using, for example, $K = 1$ examples per class in the 1-shot setting.

Contrasting the batch and stochastic settings. In the stochastic setting, access to past data is unavailable, and so none of the standard algorithms and heuristics for inference in non-parametric models are applicable [*e.g.,* 26, 25]. In particular, our proposed algorithm does not refine the cluster assignments of previously observed points by way of multiple expensive passes over the whole dataset.

In contrast, we incrementally infer model parameters and add components during episodic training based on noisy estimates of the gradients of the marginal log-likelihood. Moreover, we avoid the need to preserve task assignments, which is potentially harmful due to stale parameter values, since the task assignments in our framework are meant to be easily reconstructed on-the-fly using the `E-STEP` with updated parameters $\boldsymbol{\theta}^{(0)}, \ldots, \boldsymbol{\theta}^{(L)}, G$.

***Maximum a posteriori* estimation as iterated conditional modes.** Due to the high-dimensionality of the parameter set of neural networks, we consider a mode estimation procedure based on iterated conditional modes (ICM) [6, 59, 55, 41] that can leverage gradient computation instead of the expensive process of Gibbs sampling. iterated conditional modes (ICM) is a greedy strategy that iteratively maximizes the full conditional distribution for each variable (*i.e.,* computes the MAP estimate), instead of sampling from the conditional as is done in Gibbs sampling [55]. This leads to a fast point-estimation of the DPMM parameters in which we only need to track the means of the cluster priors.

Alternative inference procedures in probabilistic mixtures. A standard approach for estimation in latent variable models, such as probabilistic mixtures, is to represent the distribution using samples produced via some sampling algorithm.The most widely used is the Gibbs sampler [35, 17], which draws from the conditional distribution of each latent variable, given the others, until convergence to the posterior distribution over all the latents. However, in the setting of latent variables defined over high-dimensional parameter spaces such as those of neural network models, using a sampling algorithm such as the Gibbs sampler is prohibitively expensive [36, 34]. Instead of sampling, one can fit factorized variational distributions to the exact distribution $p(\phi, z|x) \approx q(\phi)q(z)$ [18, 7]. It should be noted that we do not claim that our method of point estimation in the DPMM is the most accurate method for posterior inference but we leave improved approximate inference extensions to future work.

The main drawback of using point estimates for a non-parametric mixture estimation is the inability to leverage the diffusion of the global prior G_0 when computing the likelihood of a new cluster. Highly concentrated parameter estimates for non-empty clusters should lead to low likelihoods for outlier tasks, whereas the diffused global prior should be better at capturing a wider variety of tasks.

Nonetheless, point estimation is a necessary trade-off between computation and accuracy. To allow for a more accurate estimate of the likelihood, we experimented with simulating a normal centered at the global prior mean with a variance hyperparameter that can be annealed over time to account for increased certainty about the prior choice. We can then compare the average cluster responsibility to the threshold. Another interesting extension we experimented with was to compute the gradient for each of the samples and average over the number of samples as to approximate the expectation of the gradient under the global prior. However, we found this to be less stable than simply comparing the cluster responsibilities to the threshold.

B *Maximum a posteriori* estimation in the Dirichlet process mixture model

From (4) and using a conditional mode estimate for task-specific parameters $\boldsymbol{\phi}_j$,

$$\log p\left(\boldsymbol{z}_j = \ell \mid \boldsymbol{x}_{j_{1:M}}, \boldsymbol{z}_{1:j-1}, \boldsymbol{\theta}^{(\ell)}\right) \approx \begin{cases} \log n^{(\ell)} + \log p(\boldsymbol{x}_{j_{1:M}} | \hat{\boldsymbol{\phi}}_j^{(\ell)}) + \\ \log p(\hat{\boldsymbol{\phi}}_j^{(\ell)} | \boldsymbol{\theta}^{(\ell)}) & \text{for } \ell \leq L \\ \log \zeta \ + \ \log p(\boldsymbol{x}_{j_{1:M}} | \hat{\boldsymbol{\phi}}_j^{(\ell)}) \ + \\ \log(\hat{\boldsymbol{\phi}}_j^{(\ell)} | \boldsymbol{\theta}^{(0)}) & \text{for } \ell = L+1 \ . \end{cases} \tag{5}$$

C Experimental setup

C.1 Dataset details

Few-shot regression

- Polynomial wave (Figure 4a):

$$y = \sum_i a_i x^{p_i}$$

  and $a \sim \mathcal{U}(-5.0, 5.0)$.

- Sinusoid wave (Figure 4b):

$$y = a \sin(x - \phi)$$

  where $\phi \sim \mathcal{U}(0, \pi)$ and $a \sim \mathcal{U}(0.1, 5.0)$.

- Sawtooth wave (Figure 4c):

$$y = -\frac{2a}{\pi} \arctan(\cot(\frac{x\pi}{\phi}))$$

  where $\phi \sim \mathcal{U}(0, \pi)$, $a \sim \mathcal{U}(0.1, 5.0)$.

C.2 Hyperparameter choices

C.2.1 *Mini*ImageNet few-shot classification.

We use the same data split, neural network architecture, and hyperparameter values as in [14] for common components. We use $\tau = 1$ for the softmax temperature and the same initialization as [14] for the global prior G_0. We determine an iteration number for early stopping using the validation set.

C.2.2 Continual few-shot regression.

Our architecture is a feedforward neural network with 2 hidden layers with ReLU nonlinearities, each of size 40. We use a meta-batch size of 10 tasks (both for the inner updates and the meta-gradient updates) for 5-shot regression. Our non-parametric algorithm starts with a single cluster ($L_0 = 1$ in Algorithm 3). In these experiments, we set the spawning threshold $\epsilon = 0.95T/(L+1)$, with L the number of non-empty clusters and T the size of the meta-batch. We use the mean-squared error for each task as the inner loop and meta-level objectives.

C.2.3 Continual few-shot *mini*ImageNet classification.

We use the same data split, neural network architecture, and hyperparameter values as in [14] for common components. We use a meta-batch size of 4 tasks, start with a single cluster, and set the spawning threshold to the same formula as in Section C.2.2. We use the multi-class cross entropy error for each task as the inner loop and meta-level objectives. More details on the the practical implementation for image datasets of the non-parametric algorithm can found in Section D.

D Practical and implementational details

D.1 *Task-aware vs. task-agnostic*

Since a cluster is not well-tuned immediately after its creation, we consider a cool-down period after the spawning of each new cluster where we do not consider the creation of new clusters for a fixed number of iterations, and we freeze the updating of existing clusters for a same number of iterations. This allows the newly-created cluster to take enough gradient updates in order to move from its global prior initialization, allowing it to sufficiently differentiate from the global prior.

This experimental paradigm also allows us to approximate the *task-aware* algorithms of prior work [*e.g.,* 28, 58, 38, 42] which require access to an explicit delineation between tasks that acts as a catalyst to grow model size. For the *task-aware* non-parametric mixture results reported in the experiments, we set this cool-down period to be exactly the length of the training phase for the appropriate dataset; therefore, clusters which are not meant to be specialized for the active dataset are not updated. In contrast, the *task-aware* results consider a cool-down period of $1k$ iterations, which is less than 15% of the active period for each dataset. Extensions to this fixed cool-down period could consider the rate of learning in the active cluster in order to detect when the new component has been sufficiently fit to the new task.

D.2 Practical extensions to the non-parametric algorithm

The penalty term of $\log n^{(\ell)}$ or $\log \zeta$ is necessary to regularize the likelihood of a potential new cluster in order to limit overspawning. However, in the setting where the likelihood is approximated by the loss function of a complex neural network, as in the case for most meta-learning applications, there is a large difference in orders of magnitude between the loss value (especially for the cross-entropy function) and the penalty term, even after a single batch of assignments. Furthermore, the classical log observation count $\log n$ term is misaligned with our stochastic setting for two reasons. First, since we do not re-evaluate over the whole dataset for every meta-learning episode, we are thus more concerned with the relative number of task assignments over recent iterations than the total number of assignments over the duration of training. Second, the number of tasks to be assigned can grow too large in the stochastic setting (e.g. $60k$ for *mini*ImageNet) which exacerbates the already large difference in orders of magnitudes between the loss function and the penalty term.

Accordingly, we propose two changes; First, we compute the observation based on a moving window of fixed size (5 in the experiments). Second, we apply a coefficient, which can be tuned, to the log observation count in (4). This provides more flexibility to our meta-learner as it allows it to apply to any black-box function approximator which might exhibit losses of orders of magnitudes smaller than those expected of classical probabilistic models. While the moving window size and CRP penalty coefficient terms are somewhat interdependent, we propose them as a simple starting point to tune this non-parametric meta-learner beyond what is empirically explored in this paper.

Note that without such changes in the stochastic setting of meta-learning, a nonparametric algorithm would be unable to spawn a new cluster after the first handful of iterations. Even if we were to lower the threshold ϵ, multiple almost identical clusters would be spawned in the first few iterations before it would be impossible to spawn anymore. Furthermore, the clusters would be nearly identical given the small step size of a gradient update for each meta-learning episode. Finally, this would be computationally intensive since unlike the typical applications of non-parametric mixture learning where one can afford to spawn hundreds of components then prune them over the training procedure.

D.3 Thresholding

A marked difference that is not immediate from the Gibbs conditionals is the use of a threshold on the cluster responsibilities, detailed in the `E-STEP` in Subroutine 4, to account for noise from stochastic optimization when spawning a cluster on the basis of a single batch. This threshold is necessary for the stochastic mode estimation procedure of Algorithm 3, as it ensures that a new cluster's responsibility needs to exceed a certain value before being permanently added to the set of components.

If a cluster has close to an equal share of responsibilities as compared to existing clusters after accounting for the CRP penalty $\log n^{(\ell)}$ or $\log \zeta$, it is spawned. Accordingly, this approximate inference routine still preserves the preferential attachment ("rich-get-richer") dynamics of Bayesian nonparametrics [41]. A sequential approximation for non-parametric mixtures with a similar threshold was proposed in [31] and [51], in which variational Bayes was used instead of point estimation in a DPMM.

D.4 Pruning heuristics

None of the results reported in our experiments used a pruning heuristic as we used a rather conservative hyper parameter setting that deters overspawning. We did however explore different heuristics which could work in more general settings, especially in the presence of many more latent clusters of tasks than considered in the experimental settings in this work. One such heuristic is to prune small clusters that have received disproportionately few assignments over a certain number of past iterations. Another is to evaluate the functional similarity of two clusters by computing an odds-ratio statistic for the assignment probabilities to each cluster over a set of validation tasks. If the odds-ratio statistic is below a certain threshold, the smaller cluster can be pruned.

D.5 Estimating the CRP hyperparameters

We fixed α at the size of the meta-batch. An alternative is to place a $\Gamma(1,1)$ on the concentration parameter. Based on the likelihood , the posterior is then proportional to $p(\alpha|N,K) \propto \frac{\Gamma(\alpha)}{\Gamma(\alpha+N)}\alpha^K e^{-\alpha}$ This is not a standard distribution but [39] have shown that $\log p(\alpha|N,K)$ is log-concave and methods such as L-BFGS have been used successfully in prior works. Alternatively, if we have some prior knowledge about the expected number of clusters, we can compute α based on $E[K] = \alpha \log N$. For the window-size, we considered an initial size of 20 iterations that can grow as more cluster are considered.

D.6 Implementation details

We implemented both of our parametric and non-parametric meta-learners in TensorFlow (TF) [1]. We considered 2 different settings for the `M-STEP` optimization:

- Train each cluster's parameters separately based on its corresponding loss function in an alternating manner closest to the classic EM algorithm.
- Train all cluster weights simultaneously using a surrogate loss over all validation batches.

Since the latter better leverages the differentiability of softmax-clustering and performed better empirically, we used it to report all experimental results.

D.6.1 Nonparametric Implementation

For the nonparametric algorithm, we chose the first approach to the `M-STEP` by constructing separate optimizers for each cluster's parameters. We pre-allocate a set of weights and use a mask during training to discard the parameters of empty clusters due to the static nature of TF graphs. When the algorithm exhausts the set of pre-allocated weights, we simply construct more network weight and reinitialize our optimizers.

D.6.2 CRP global prior

The likelihood of a new cluster is sensitive to the choice of a base measure or prior prior, G_0 on the cluster hyperparameters. Our gradient-based point estimation does not make any modeling assumption on the distribution of the weights, rendering the problem of principally updating the base measure, after or during training, non-trivial. We chose to initialize all weights with zero-mean normals in the fully-connected layers. For the convolutional layers, we leveraged Xavier initialization [19] similarly to prior work [14] in meta-learning.

However, such initialization is poor in the non-parametric for most non-trivial regression or classification tasks. Therefore, in the nonparametric setting, we start with a single cluster for a fixed number of iterations. We then initialize all clusters with the weights of the first clusters. This set of weights can be considered as the mean of the base measure or global prior in our setting.

We periodically update the global prior using a uniform average of the parameters of the existing clusters. This can be done by simply averaging over the parameter of the non-empty clusters as weighted by their sizes. Note that, we found that performing weighted KDE smoothing with a small bandwidth hyperparameter to perform slightly better than the average which is to be expected for neural network parameters. The number of iterations between updates of the global prior is a hyperparameter that we tune on the validation set. It is also possible to continuously, but less frequently over time, update this global prior as more data is encountered.

References

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[2] B. Bakker and T. Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, 4(May):83–99, 2003.

[3] M. Bauer, M. Rojas-Carulla, J. B. Świkatkowski, B. Schölkopf, and R. E. Turner. Discriminative k-shot learning using probabilistic models. *arXiv preprint arXiv:1706.00326*, 2017.

[4] J. Baxter. A bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine learning*, 28(1):7–39, 1997.

[5] J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12(1):149–198, 2000.

[6] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 259–302, 1986.

[7] D. M. Blei, M. I. Jordan, et al. Variational inference for dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[8] R. Caruana. Multitask learning: A knowledge-based source of inductive bias. In *Proceedings of the 10th International Conference on Machine Learning (ICML)*, 1993.

[9] R. Caruana. Multitask learning. In *Learning to learn*, pages 95–133. Springer, 1998.

[10] A. G. Collins and M. J. Frank. Cognitive control over learning: Creating, clustering, and generalizing task-set structure. *Psychological review*, 120(1):190, 2013.

[11] H. Daumé III. Bayesian multitask learning with latent hierarchies. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 135–142, 2009.

[12] T. Deleu and Y. Bengio. The effects of negative adaptation in model-agnostic meta-learning. *arXiv preprint arXiv:1812.02159*, 2018.

[13] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 1–38, 1977.

[14] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[15] C. Finn, K. Xu, and S. Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pages 9516–9527, 2018.

[16] J. Gao, W. Fan, J. Jiang, and J. Han. Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 283–291. ACM, 2008.

[17] S. J. Gershman and D. M. Blei. A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1–12, 2012.

[18] Z. Ghahramani and M. J. Beal. Variational inference for bayesian mixtures of factor analysers. In *Advances in neural information processing systems*, pages 449–455, 2000.

[19] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[20] J. Gordon, J. Bronskill, M. Bauer, S. Nowozin, and R. Turner. Meta-learning probabilistic inference for prediction. In *ICLR*, 2019.

[21] E. Grant, C. Finn, S. Levine, T. Darrell, and T. Griffiths. Recasting gradient-based meta-learning as hierarchical Bayes. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*, 2018.

[22] K. Greff, S. van Steenkiste, and J. Schmidhuber. Neural expectation maximization. In *Advances in Neural Information Processing Systems*, pages 6694–6704, 2017.

[23] S. Gupta, D. Phung, and S. Venkatesh. Factorial multi-task learning: a bayesian nonparametric approach. In *International conference on machine learning*, pages 657–665, 2013.

[24] T. Heskes. Solving a huge number of simular tasks: a combination of multi-task learning and a hierarchical bayesian approach. 1998.

[25] M. C. Hughes, E. Fox, and E. B. Sudderth. Effective split-merge monte carlo methods for nonparametric models of sequential data. In *Advances in neural information processing systems*, pages 1295–1303, 2012.

[26] S. Jain and R. M. Neal. A split-merge markov chain monte carlo procedure for the dirichlet process mixture model. *Journal of computational and Graphical Statistics*, 13(1):158–182, 2004.

[27] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[28] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[29] N. D. Lawrence and J. C. Platt. Learning to learn with the informative vector machine. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, page 65, 2004.

[30] Y. Lee and S. Choi. Gradient-based meta-learning with learned layerwise metric and subspace. In *ICML*, 2018.

[31] D. Lin. Online learning of nonparametric mixture models via sequential variational approximation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 395–403, 2013.

[32] D. MacKay. A practical Bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

[33] N. Metropolis and S. Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

[34] P. Müller and D. R. Insua. Issues in bayesian analysis of neural network models. *Neural Computation*, 10(3):749–770, 1998.

[35] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265, 2000.

[36] R. M. Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[37] R. M. Neal and G. E. Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pages 355–368. Springer, 1998.

[38] C. V. Nguyen, Y. Li, T. D. Bui, and R. E. Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

[39] C. E. Rasmussen. The infinite gaussian mixture model. In *Advances in neural information processing systems*, pages 554–560, 2000.

[40] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, 2017.

[41] Y. P. Raykov, A. Boukouvalas, F. Baig, and M. A. Little. What to do when k-means clustering fails: a simple yet principled alternative algorithm. *PloS one*, 11(9):e0162259, 2016.

[42] H. Ritter, A. Botev, and D. Barber. Online structured Laplace approximations for overcoming catastrophic forgetting. In *Advances in Neural Information Processing Systems*, pages 3738–3748, 2018.

[43] H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, 22(3):400–407, 1951.

[44] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich. To transfer or not to transfer. In *NIPS 2005 workshop on transfer learning*, volume 898, pages 1–4, 2005.

[45] A. J. Rothman, E. Levina, and J. Zhu. Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, 19(4):947–962, 2010.

[46] R. Salakhutdinov, J. B. Tenenbaum, and A. Torralba. Learning with hierarchical-deep models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1958–1971, 2013.

[47] J. Schmidhuber. *Evolutionary principles in self-referential learning*. PhD thesis, Institut für Informatik, Technische Universität München, 1987.

[48] D. Sculley. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM, 2010.

[49] J. Snell, K. Swersky, and R. S. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NIPS) 30*, 2017.

[50] N. Srivastava and R. R. Salakhutdinov. Discriminative transfer learning with tree-based priors. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2094–2102, 2013.

[51] A. Tank, N. Foti, and E. Fox. Streaming variational inference for bayesian nonparametric mixture models. In *Artificial Intelligence and Statistics*, pages 968–976, 2015.

[52] S. Thrun. Discovering structure in multiple learning tasks: The tc algorithm.

[53] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NIPS) 29*, pages 3630–3638, 2016.

[54] J. Wan, Z. Zhang, J. Yan, T. Li, B. D. Rao, S. Fang, S. Kim, S. L. Risacher, A. J. Saykin, and L. Shen. Sparse Bayesian multi-task learning for predicting cognitive outcomes from neuroimaging measures in Alzheimer's disease. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 940–947. IEEE, 2012.

[55] M. Welling and K. Kurihara. Bayesian k-means as a "maximization-expectation" algorithm. In *Proceedings of the 2006 SIAM international conference on data mining*, pages 474–478. SIAM, 2006.

[56] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(Jan):35–63, 2007.

[57] K. Yu, V. Tresp, and A. Schwaighofer. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 1012–1019, 2005.

[58] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. *arXiv preprint arXiv:1703.04200*, 2017.

[59] Y. Zhang, M. Brady, and S. Smith. Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE transactions on medical imaging*, 20(1):45–57, 2001.

[60] Y. Zhang and J. G. Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems*, pages 2550–2558, 2010.

[61] Y. Zhang and D.-Y. Yeung. A regularization approach to learning task relationships in multitask learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(3):12, 2014.