`from domain import knowledge:` Enhancing Language Models for Technical Domains with Dynamic Token Injection

Giorgio Giannone
Microsoft Research
Technical University of Denmark
gigi@dtu.dk

Neil Tenenholtz
Microsoft Research
netenenh@microsoft.com

James Hall
Microsoft Research
jamhall@microsoft.com

Nicolo Fusi
Microsoft Research
fusi@microsoft.com

David Alvarez Melis
Microsoft Research
Harvard University
dam@seas.harvard.edu

Abstract

Large language models (LLMs) are rapidly advancing the frontier of natural language understanding and generation. Their generalist nature, while adept at handling a wide range of tasks, often lacks the depth and precision required by highly specialized and rapidly evolving technical domains, such as genomics and engineering design. Fine-tuning these models for specific domains can be effective but requires large amounts of data and compromises their general reasoning capabilities. In this work, we introduce a scalable method to infuse specialized knowledge into generalist language models by dynamically extending their vocabulary with specialist tokens. By using a lightweight functional mapping on an extended vocabulary and adjusting the logit distribution, we enable the model to grasp domain-specific nuances. We demonstrate this in an application in genomics, where we extend a standard LLM by introducing knowledge about a large set of genes, allowing it to proficiently tackle tasks involving both textual and genetic data. Functional alignment enables the model to handle novel gene tokens that were never encountered during training, enabling domain-aware out-of-distribution capabilities in generalist language models.

1 Introduction

In recent years, natural language processing has witnessed significant advancements driven by the advent of large language models (LLMs, [5, 17, 2]). LLMs have proven to be flexible multitask generative models, capable of adapting to new tasks and scenarios leveraging in-context learning [2] and prompting [20]. However, in numerous scientific and engineering scenarios, there arises a crucial requirement for domain-specific models capable of dealing with intricate concepts. Take genomics, for instance, where we have thousands of genes that can be expressed in diverse manners —–through sequences, names, symbols, and functionality—– and are entwined within a sophisticated intra-domain structure. In such a field, a language-capable model proficient in handling technical information would prove exceptionally valuable.

While prompting and in-context learning can help, such methods are expensive or require large models to be effective. Conversely, fine-tuning alters the base model, creating a domain-specific model that is limited by the expressivity of the finetuning set, making generalizing to out-of-distribution concepts especially challenging. Fine-tuning may also diminish the model's overall capability,

NeurIPS 2023 Generative AI and Biology Workshop.

leading to interference between the two regimes. As an illustration, consider the letter `C` in a general LLM, where it may be closely associated with words like `Computer`. However, in a specialized language model for chemistry, `C` represents a symbol with a high-level meaning related to `Carbon`. When introducing abstract concepts characterized by complex meanings, such as genes, our goal is to seamlessly integrate these concepts into the LLM. Our particular focus is on scenarios where these rich concepts originate externally, in domains unknown or underrepresented in the LLM's training data, such as genomics and chemistry. We aspire to utilize general LLMs as robust reasoning engines, benefiting from their extensive training data, but making them incorporate these structure-rich concepts, enabling dynamic expansion of the domain knowledge when new information emerges (e.g., new genes, pathways, or connections).

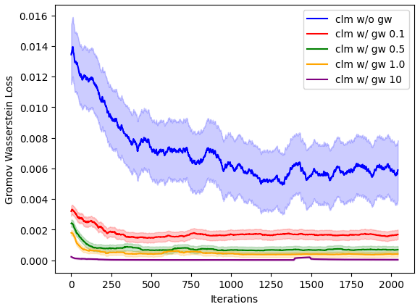

Figure 1: Gromov-Wasserstein loss (Eq. 3) on in- distribution genes. Our approach tends naturally to align the specialized domain before and after adaptation. Increasing the GW loss contribution, the fea- tures tend to be closer and closer, enforcing full preservation of the specialized structure.

Can we introduce specialized tokens with rich structures in a general language model and modify a minimal set of parameters? We answer this question in the affirmative, by showing how this challenging task can be accomplished using frozen LLMs, appropriate initialization, normalization techniques, alignment of augmented embeddings, and adapting augmented likelihood through a shared set of learnable weights.

Contribution. Our contributions are the following: **(i)** We propose a scalable method to adapt and align a frozen general language model with a specialized domain, injecting specialist external knowledge into a general model. **(ii)** We introduce a functional mapping to augment the vocabulary in a dynamic fashion, enabling out-of-distribution generalization to new concepts. **(iii)** We empirically show that our model solves tasks involving general and specialized domains as well as aligns specialized knowledge before and after adaptation.

2 Background

Domain-specialized Language Models. Language Models [5, 17, 2] excel in generating data of high diversity for unstructured domains. However, in many specialized domains factuality and constraint satisfaction are important, and smaller specialized models, which are easy to deploy and inexpensive to sample, are needed. In particular, LLMs have been employed in the life sciences to translate between text and chemistry [4, 6], biology [12], medicine [19], DNA-sequencing [15], and protein sequences [10, 13] and folding [8].

Adaptation Techniques for Language Models. Fine-tuning involves training a pre-existing language model on domain-specific data or tasks [5]. This process allows the model to adapt to the nuances of the target domain, making it more proficient in domain-specific tasks. Efficient fine-tuning methods [7] can be leveraged to greatly decrease the learned parameter count maintaining performance at the cost of additional data and hyperparameter tuning. In-context learning [2] aims to improve the capabilities of a model by providing additional information or examples in the prompt. Prompt-tuning [9] techniques involve the introduction of virtual tokens or placeholders within the text. These tokens can be used to guide the model's attention and understanding of specific concepts or entities. Adapting general transformers for specific tasks has been proposed in [18], where pretrained language models are leveraged to solve novel tasks using distributional alignment and fine-tuning, extending the ideas introduced in [11].

3 Method

We consider autoregressive language models consisting of a weight-tied embedding layer $\mathbf{e}$, a $\texttt{LLM}(;\phi)$ backbone comprised of transformer blocks, a language modeling head whose weights are shared with the embedding layer, and then a softmax function σ computes the output distribution. Given a general autoregressive language model [16]), standard causal language modeling can be written as:

$$\mathcal{L}_{\texttt{CLM}}(\mathbf{e}_g) = -\mathbf{x}_{[1:]} \log p(\mathbf{x}_{[:-1]}), \quad p(\mathbf{x}) = \sigma(\mathbf{h}(\mathbf{x})\, \mathbf{e}_g^T), \quad \mathbf{h}(\mathbf{x}) = \texttt{LLM}(\mathbf{e}_g\,[\mathbf{x}]\,;\phi), \tag{1}$$

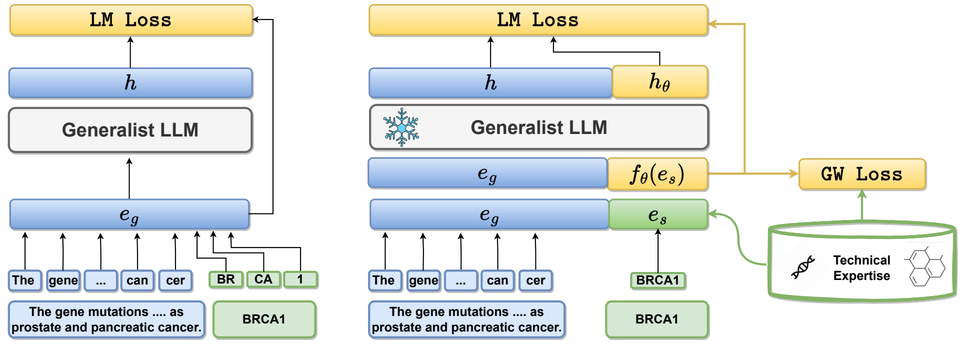

Figure 2: Alignment and Adaptation Pipeline. We inject a new domain using a large set of specialized tokens (gene names), augmenting the general vocabulary by up to 30%. These tokens are initialized from a domain-specialized model. We then align the specialized embeddings and logit distribution with a lightweight adapter layer. We train the adapter network with pairs of (text, gene) using standard causal language modeling.

where $\mathbf{e}_g \in \mathbb{R}^{g \times d}$ is the embedding layer, $\mathbf{x} \in \mathbb{R}^{t \times d}$ is a sequence of t indexes that index the embedding layer, $\mathbf{h} \in \mathbb{R}^{t \times d}$ is the last hidden state, and σ is the softmax function. Notice how the embeddings weights, $\mathbf{e}_g$ have dimensions of the vocabulary size g and the hidden size d, are transposed in the output of the model and used to compute the logits together with the last hidden states. Now, our goal is to inject new knowledge into the LLM by adding a large set of specialized tokens with abstract meaning attached to them (e.g., concepts, gene names) without finetuning the core model, instead adapting the embedding and logits layer to align the novel, specialized vocabulary to the base, general one. This can be achieved by leveraging the following formulation, where f_θ is an embedding mapping function and $\mathbf{e}_\theta = [\mathbf{e}_g,\ f_\theta(\mathbf{e}_s)]$ is the augmented vocabulary:

$$\mathcal{L}_{\texttt{CLM}}(\mathbf{e}_g, \mathbf{e}_s, \theta) = -\mathbf{x}_{[1:]} \log p_\theta(\mathbf{x}_{[:-1]}), \quad p_\theta(\mathbf{x}) = \sigma(\mathbf{h}^\theta(\mathbf{x})\ \mathbf{e}_\theta^T), \quad \mathbf{h}^\theta(\mathbf{x}) = \texttt{LLM}(\mathbf{e}_\theta\,[\mathbf{x}]\,;\ \bar{\phi}). \quad (2)$$

Here the embedding layer is augmented with s tokens using $\mathbf{e}_s \in \mathbb{R}^{s \times d'}$, in general of different dimensionality than d. Then the specialized embeddings are aligned and adapted leveraging a small network. In particular a linear layer $W \in \mathbb{R}^{d' \times d}$ is used for dimensionality alignment, and then a ResNet block f_θ takes in input $\mathbf{e}_s W$ and output the adapted embeddings $\mathbf{e}_\theta = [\mathbf{e}_g, f_\theta(\mathbf{e}_s W_\theta)]$. The augmented embeddings will have dimension $((g + s) \times d)$.

Functional Alignment. Conceptually, adaptation involves a two-step procedure: dimensionality adaptation, namely projecting the external knowledge embeddings into a space of the same dimensionality as the model's embeddings, and domain adaptation, which consists of geometrically aligning the general and specialized domains within a single model with minimal intervention. Once learned, the function f_θ, provides a versatile mapping that aligns the augmented embedding layer with the general model. Simultaneously, we employ the same set of weights to adjust the logit distribution, as depicted in Figure 2.

The Gromov-Wasserstein distance [14, 1, 3] provides a notion of dissimilarity between two metric spaces based on the correspondence of pairwise relationships within these spaces. In contrast to traditional notions of distance between point clouds and distributions that rely entirely on pairwise distances across collections, GW compares collections based on distances *within* each of the spaces, allowing for comparison across spaces that are not a-priori aligned (e.g., of different dimensionality). This makes GW an ideal metric for comparing datasets from heterogeneous domains, such as those encountered in the fields of biology and genomics. In formula:

$$\mathcal{L}_{\texttt{GW}}(d_g, d_s; \Gamma) = \min_{\Gamma \in (p,q)} \sum_{ijkl} L(d(\mathbf{x}_g^i, \mathbf{x}_g^k), d'(\mathbf{x}_s^j, \mathbf{x}_s^l))\Gamma_{ij}\Gamma_{kl} = \min_{\Gamma \in (p,q)} \sum_{ijkl} L_{ijkl}\Gamma_{ij}\Gamma_{kl}, \quad (3)$$

where d and d' are (potentially different) metrics, L is a loss function between pairs of distances (e.g., L2 loss) so that L_{ijkl} measures the similarity between pair-wise distances across the two domains, Γ_{ij} is a coupling between item i in one domain and object j in the other, and $p = \Gamma 1_n$ and $q = \Gamma^T 1_n$. Thus, problem (3) seeks a coupling Γ between the two distributions that minimize the total alignment cost between them, or equivalently, that maximizes their geometric agreement. In the following, we will use this metric to quantify alignment pre- and post-adaptation (Fig. 1).

4 Experiments

Setup. For all the experiments, we freeze the base model and augment the embedding layers with 17k tokens, each one representing one gene (10k used for training and the remaining for evaluation). We extract a set of genes and their relative descriptions as described in [21]. We use a small one-layer ResNet to align the specialized embeddings with the general embeddings, and use the same weights transposed for the likelihood (recall that we consider weight-tied models). We initialize the specialized embeddings using the hidden states of biogpt-large [12], a causal language model specialized in biology. We normalize the specialized embeddings such that the norm of the general and specialized embeddings are of the same average magnitude before finetuning. We then fine-tune the functional mapping (with the LLM backbone frozen) on a small set of paired descriptions and genes (see Fig. 3), over-weighting the cross-entropy loss by a factor of two over the specialized vocabulary. All models are trained and evaluated on a single A100.

Evaluation. We evaluate our method on a task that we call `text2gene`: this task consists of providing a textual description of a gene function and asking the model to choose between the correct gene and random ones using the likelihood as a classification signal. Our goal is to understand if the general vocabulary, which processes general text, and the specialized vocabulary, which processes the genes, are aligned and can provide meaningful results in a language model. For the quantitative evaluation, we consider binary and multiclass classification (Table 1), and top-k retrieval (Fig. 4). For the qualitative evaluation, we measure the similarity between a common gene (`BRCA1`) before and after adaptation (Fig. 5 and 6). We also plot the top-10 logits for a given target gene conditioning on a description (Table 2). In Fig. 1 we show alignment pre- and post-adaptation, as measured by the Gromov-Wasserstein (GW) loss, with different levels of strength for the GW loss.

Table 1: Classification accuracy leveraging the adapted likelihood $p_\theta(\mathbf{x})$ as signal for 2 and 10 classes. `IN`: genes are seen during training. `OOD`: genes not seen during training. We see that by leveraging the augmented vocabulary and the adapted likelihood we can solve classification tasks for a specialized task. Leveraging the functional approach, we are also able to deal with genes never seen during training.

								`text2gene`	
Model	Size	Params	Loss	GW	Nt	Init	Adapter	Train (`IN`)	Val (`OOD`)
random (2 classes)	-	-	-	-	-	-	-	0.500	
biogpt-l	1.5B	-	clm	-	-	-	-	0.801	
gpt2-xl	1.5B	-	clm	-	-	-	-	0.710	
gpt2-xl	1.5B	3.8M	clm	0.0	10k	hidden	mlp	0.994	0.930
gpt2-xl	1.5B	3.8M	clm-gw	0.1	10k	hidden	mlp	0.996	**0.934**
gpt2-xl	1.5B	3.8M	clm-gw	0.5	10k	hidden	mlp	0.993	**0.937**
random (10 classes)	-	-	-	-	-	-	-	0.100	
gpt2-xl	1.5B	3.8M	clm	0.0	10k	hidden	mlp	0.99	0.755
gpt2-xl	1.5B	3.8M	clm-gw	0.1	10k	hidden	mlp	0.988	0.763
gpt2-xl	1.5B	3.8M	clm-gw	0.5	10k	hidden	mlp	0.985	**0.772**

Results. Table 1 shows the classification accuracy of different models utilizing the adapted likelihood $p_\theta(\mathbf{x})$, specifically for binary and 10-way classification problems. The baselines have different initialization, adaptation strategy, and alignment weighting. For further baselines, see Table 3 in the appendix. We measure performance for genes seen during training (`IN`) and those not encountered during training (`OOD`). All target genes leverage the augmented vocabulary, i.e. each gene is represented as a technical token. The gpt2-xl model, when adapted with a small ResNet block and aligned using the gw loss with different strength levels, consistently performs well for in-distribution genes. Interestingly, the model with adaptation performs well on out-of-distribution genes (genes never seen during training), peaking at approximately 93.7% for binary and 77.2% for 10-way classification. These results demonstrate the effectiveness of our adaptation strategy in enhancing vocabulary and likelihood. By aligning external knowledge, we harness the general reasoning capabilities of our generalist model to assign a higher likelihood to the correct gene description. Notably, our functional approach can handle out-of-distribution scenarios, where conventional fine-tuning of embeddings would struggle without the application of extrapolation techniques and retrieval methodologies. In Figure 4, we present the top-k accuracy results for binary classification tasks involving both in-distribution and out-of-distribution genes, both with and without GW loss. These findings reveal that our likelihood calibration is remarkably accurate for in-distribution genes, and it demonstrates the capability to identify relevant genes even in the more challenging out-of-distribution context. This additional experimental evidence further validates the effectiveness of our approach in integrating domain knowledge into a generalist language model.

5 Conclusion

We introduce a method that combines domain knowledge with the reasoning of generalist LLMs, using a dynamic vocabulary and functional mapping to adapt the likelihood distribution. This promising approach could hold the key to unlocking rich, highly structured, specialized domain datasets for use within an LLM setting. Where in this work we focus on genes and biology, we expect such a method to be of potential applicability broadly in natural and engineering sciences.

References

[1] D. Alvarez-Melis and T. Jaakkola. Gromov-Wasserstein alignment of word embedding spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] C. Bunne, D. Alvarez-Melis, A. Krause, and S. Jegelka. Learning generative models across incomparable spaces. In *International conference on machine learning*, pages 851–861. PMLR, 2019.

[4] D. Christofidellis, G. Giannone, J. Born, O. Winther, T. Laino, and M. Manica. Unifying molecular and textual representations via multi-task language modelling. *arXiv preprint arXiv:2301.12586*, 2023.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[6] C. Edwards, C. Zhai, and H. Ji. Text2mol: Cross-modal molecule retrieval with natural language queries. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 595–607, 2021.

[7] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[8] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[9] B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[10] Z. Lin, H. Akin, R. Rao, B. Hie, Z. Zhu, W. Lu, N. Smetanin, R. Verkuil, O. Kabeli, Y. Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[11] K. Lu, A. Grover, P. Abbeel, and I. Mordatch. Frozen pretrained transformers as universal computation engines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7628–7636, 2022.

[12] R. Luo, L. Sun, Y. Xia, T. Qin, S. Zhang, H. Poon, and T.-Y. Liu. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6):bbac409, 2022.

[13] A. Madani, B. McCann, N. Naik, N. S. Keskar, N. Anand, R. R. Eguchi, P.-S. Huang, and R. Socher. Progen: Language modeling for protein generation. *arXiv preprint arXiv:2004.03497*, 2020.

[14] F. Mémoli. Gromov–wasserstein distances and the metric approach to object matching. *Foundations of computational mathematics*, 11:417–487, 2011.

[15] E. Nguyen, M. Poli, M. Faizi, A. Thomas, C. Birch-Sykes, M. Wornow, A. Patel, C. Rabideau, S. Massaroli, Y. Bengio, et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. *arXiv preprint arXiv:2306.15794*, 2023.

[16] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[17] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

[18] J. Shen, L. Li, L. M. Dery, C. Staten, M. Khodak, G. Neubig, and A. Talwalkar. Cross-modal fine-tuning: Align then refine. *arXiv preprint arXiv:2302.05738*, 2023.

[19] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, and D. S. W. Ting. Large language models in medicine. *Nature medicine*, pages 1–11, 2023.

[20] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[21] Y. Zhang, Q. Chen, Y. Zhang, Z. Wei, Y. Gao, J. Peng, Z. Huang, W. Sun, and X.-J. Huang. Automatic term name generation for gene ontology: task and dataset. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4705–4710, 2020.

A Additional Experiments

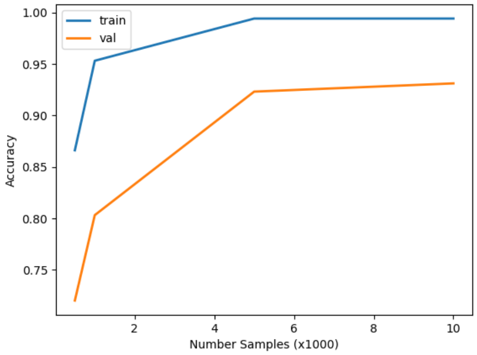

Figure 3: Binary Accuracy on in-distribution (train) and out-of-distribution (val) genes increasing the number of genes used during training, from 500 to 10000. Performance improves monotonically with more data and plateaus around 5000 samples.

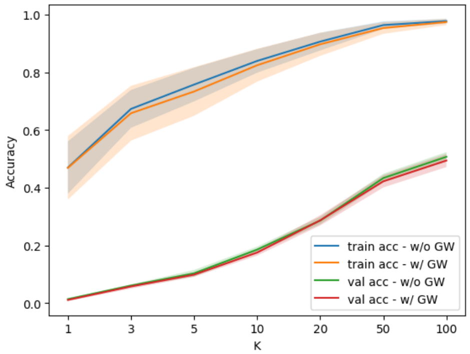

Figure 4: top-k accuracy for in- and out-of-distribution evaluation sets. The in-distribution text description is novel and genes are seen during training. For the out-of-distribution, text and genes are novel. We see how the likelihood is relatively good at retrieving the relevant genes. See also 2.

Table 2: Top-10 logits (over the full vocabulary) for gene prediction on in-distribution genes. We see that, given a target, the likelihood assigns mass in the neighbors of the correct gene, providing qualitative evidence that the general model is able to deal with the specialized knowledge extracting the structure of the novel domain, and not only memorizing the input/output mappings.

Target	NME4	APOL1	APOL3	SERPINA5	APOD	APOA4
logit 1	NME3	**APOL1**	APOL1	SERPINB4	LIPC	APOOL
logit 2	**NME4**	APOL2	APOL2	SERPINB12	GPIHBP1	APCS
logit 3	NME5	APOL5	APOLD1	SERPINB13	APOA2	APOO
logit 4	NME8	APOA5	APOL5	SERPINB10	APOC1	APOA2
logit 5	NME9	APOA1	**APOL3**	SERPINB11	**APOD**	APOC3
logit 6	NME7	APOL3	APOA5	SERPINA12	APOO	ATIC
logit 7	NME6	APOC3	APOL6	SERPINA1	APOL2	APOC4
logit 8	NME2	ABCA1	APOA4	SERPINA3	APOL6	APOC1
logit 9	NME2P1	APOLD1	APOA2	SERPINB6	APOL4	APOLD1
logit 10	NMES1	APOL6	APOA1	SERPINB3	APOE	ADIPOQ

Table 3: Results on binary classification leveraging the adapted likelihood $p_\theta(\mathbf{x})$. * For gpt-3.5 we do not have access to the likelihood, and we constrain the output to one token and [0, 1] for the answer. In-distribution genes are seen during training, whereas out-of-distribution genes not seen during training. clm: causal language modeling loss. ft: fine-tuning with supervised fine-tuning and RLHF. gw: gromov-wasserstein loss.

								`text2gene`	
Model	Size	Params	Loss	GW	Nt	Init	Adapter	Train (in)	Val (out)
2 classes									
random	-	-	-	-	-	-	-	0.500	
gpt2-xl	1.5B	-	clm	-	-	-	-	0.710	
biogpt-l	1.5B	-	clm	-	-	-	-	0.801	
llama2	7B	-	clm-ft	-	-	-		0.873	
llama2	13B	-	clm-ft	-	-	-		0.865	
gpt-3.5*	100B+	-	clm-ft	-	-	-	in-context	0.897	
gpt2-xl	1.5B	2.5M	clm	0.0	10k	hidden	linear	0.537	0.511
gpt2-xl	1.5B	2.5M	clm-gw	0.1	10k	hidden	linear	0.523	0.520
gpt2-xl	1.5B	3.8M	clm	0.0	10k	random	mlp	0.530	0.517
gpt2-xl	1.5B	3.8M	clm-gw	0.1	10k	random	mlp	0.533	0.505
gpt2-xl	1.5B	3.8M	clm	0.0	10k	hidden	mlp	0.994	0.930
gpt2-xl	1.5B	3.8M	clm-gw	0.1	10k	hidden	mlp	0.996	**0.934**
gpt2-xl	1.5B	3.8M	clm-gw	0.5	10k	hidden	mlp	0.993	**0.937**
gpt2-xl	1.5B	3.8M	clm-gw	1.0	10k	hidden	mlp	0.979	0.933
10 classes									
random	-	-	-	-	-	-	-	0.100	
gpt2-xl	1.5B	3.8M	clm	0.0	5k	hidden	mlp	0.989	0.690
gpt2-xl	1.5B	3.8M	clm-gw	0.1	5k	hidden	mlp	0.987	0.714
gpt2-xl	1.5B	3.8M	clm-gw	0.5	5k	hidden	mlp	0.983	0.611
gpt2-xl	1.5B	3.8M	clm-gw	1.0	5k	hidden	mlp	0.983	0.698
gpt2-xl	1.5B	3.8M	clm	0.0	10k	hidden	mlp	0.99	0.755
gpt2-xl	1.5B	3.8M	clm-gw	0.1	10k	hidden	mlp	0.988	0.763
gpt2-xl	1.5B	3.8M	clm-gw	0.5	10k	hidden	mlp	0.985	**0.772**
gpt2-xl	1.5B	3.8M	clm-gw	1.0	10k	hidden	mlp	0.989	0.739

B Qualitative Results

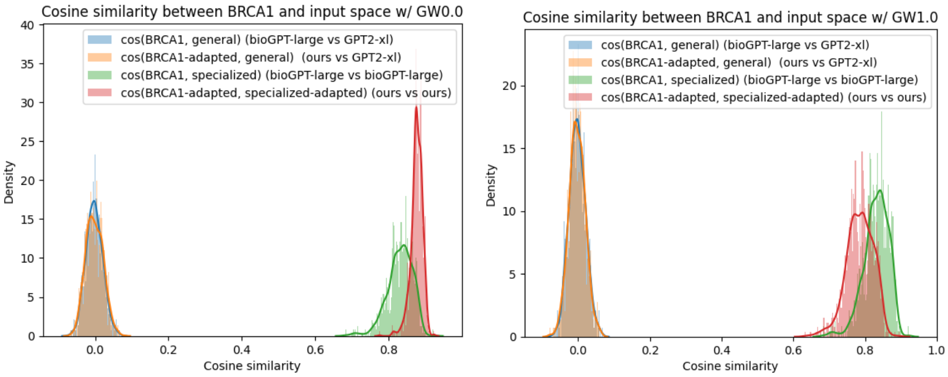

Figure 5: Input Alignment. Cosine Similarity between the general domain (GPT2 embeddings) and a specialized gene name (BRCA1) w/ and w/o adaptation.

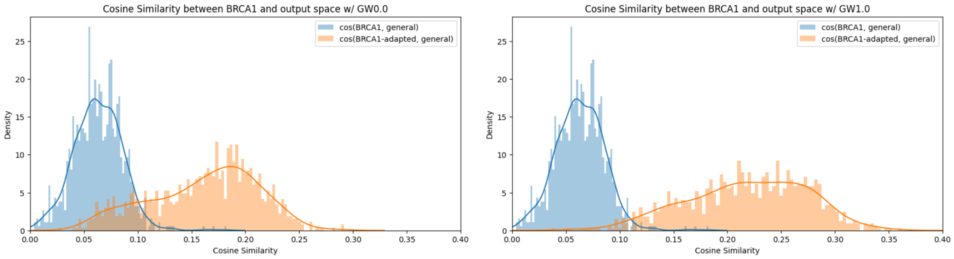

Figure 6: Output Alignment. Cosine Similarity between the general domain (GPT2 hidden states) and a specialized gene name (BRCA1) w/ and w/o adaptation.

C Dataset

Table 4: text2gene prompt structure.

	Prompt	Description
Task	`text2gene:`	task prompt
Text	`<input_text>`	input text
Domain	`<|GENE|>`	domain
Gene	`<SPECIALIZED_TOKEN>_<target_gene>`	The target gene
End Token	`<|END|>`	end

D Details

Table 5: Relevant Hyperparameters for all the models and GPT2 baselines. CE: Cross-Entropy. GW: Gromov-Wasserstein.

Key	Value
Batch size	64
Architecture	gpt2-xl
Epochs	50-100
Learning rate	$3e^{-4}$
Loss	CE
Alignment	GW
λ	0.1-10.0
Optimizer	AdamW
Initialization	biogpt-large
Normalization	norm-based
Re-weighting ratio	1:2
Vocabulary size	50257
Technical Tokens (Train/Val)	17000 (10000/7000)