Time is Encoded in the Weights of Finetuned Language Models

Kai Nylund[1] **Suchin Gururangan**[1] **Noah A. Smith**[1,2]
[1]Paul G. Allen School of Computer Science & Engineering, University of Washington
[2]Allen Institute for AI
knylund@cs.washington.edu

Abstract

We present *time vectors*, a simple tool to customize language models to new time periods. Time vectors are created by finetuning a language model on data from a single time (e.g., a year or month), and then subtracting the weights of the original pretrained model. This vector specifies a direction in weight space that, as our experiments show, improves performance on text from that time period. Time vectors specialized to adjacent time periods appear to be positioned closer together in a manifold. Using this structure, we interpolate between time vectors to induce new models that perform better on intervening and future time periods, without any additional training. We demonstrate the consistency of our findings across different tasks, domains, model sizes, and time scales. Our results suggest that time is encoded in the weight space of finetuned models.

1 Introduction

Temporal variation is a fundamental characteristic of language. As we show in §3, it manifests in language model development as *temporal misalignment*, where deviations in train and test data lead to large performance degradation across different time periods (Luu et al., 2022; Lazaridou et al., 2021, *inter alia*). This necessitates adaptation techniques for customizing models to specific time periods. Designing such techniques is difficult, however, due to the multitude of time scales and the possibility that data from a target time period might be unavailable.

Recent work has shown that the behavior of neural networks can be edited through closed-form interpolation between parameters of finetuned models (Ilharco et al., 2023; Ortiz-Jiménez et al., 2023; Li et al., 2022; Wortsman et al., 2021, *inter alia*). In this work, we demonstrate that weight-space interpolation can also be used to cheaply edit language model behavior over *time*. To this end, we introduce *time vectors* (§4), an extension of task vectors (Ilharco et al., 2023). We finetune a pretrained language model on text from a single time period, and then subtract the pretrained weights. This vector represents a direction of movement in weight space that improves performance on text from the target time period.

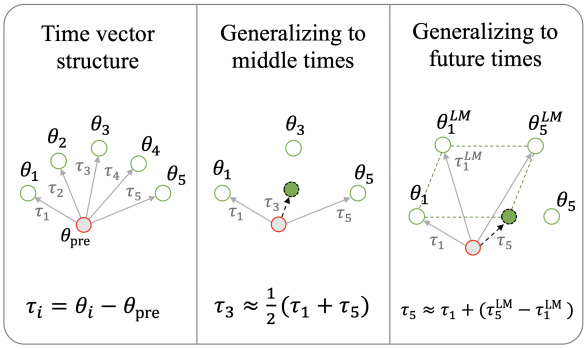

Figure 1: **We present *time vectors*, a simple tool to customize language models to new time periods.** Time vectors (τ_i) specify a direction in weight space that improves performance on text from a time period i. They are computed by subtracting the pretrained weights (θ_{pre}; left panel) from those finetuned to a target time period (θ_i). We can customize model behavior to new time periods (e.g., intervening months or years) by interpolating between time vectors and adding the result to the pretrained model (middle panel). We can also generalize to a future time period j with analogy arithmetic (right panel). This involves combining a task-specific time vector with analogous time vectors derived from finetuned language models (τ_j^{LM}).

We analyze the structure of time vectors with temporally organized datasets for language modeling, classification, and summarization (§2). Our results consistently suggest that time vectors are intuitively organized on a manifold; years or months that are closer together in time yield time vectors that are also closer together in weight space. Similarly, we show that temporal degradation in yearly and monthly settings is strongly correlated with the angles between time vectors (§4.2).

We use this structure of time vectors to induce

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2571–2587

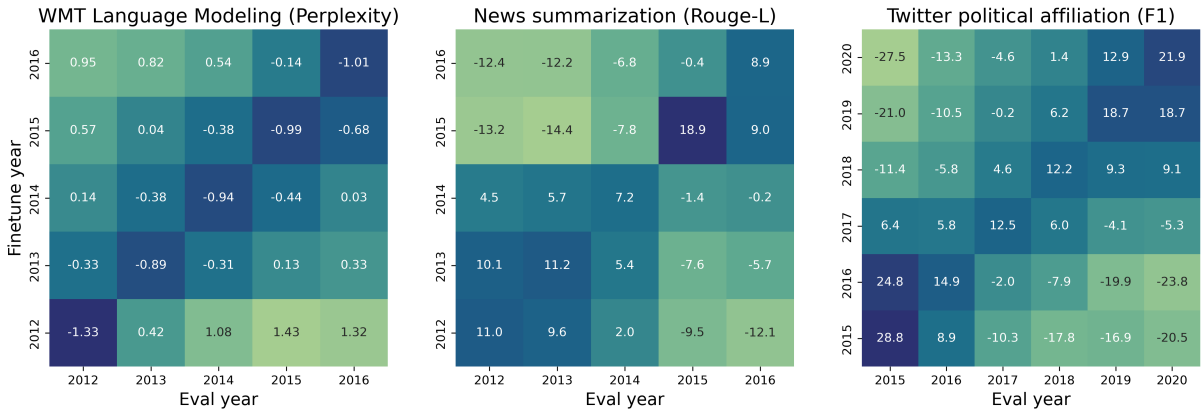

Figure 2: **Model performance degrades linearly year-to-year.** We evaluate language model perplexity (WMT), ROUGE-L (news summarization), and macro F1 (political affiliation classification). Each cell indicates the monthly performance of T5-3B finetuned and evaluated on a *single year* from that task. We report the percentage difference from the average performance for each year, and find linear degradation as finetuning and evaluation years become more misaligned regardless of task. We display similar trends for T5-small and medium, as well as for other domains and tasks, in §A.1. We measure the linearity of these degradations in Appendix Table 4.

models that generalize better to data from new time periods. By interpolating between two time vectors, we discover vectors that, when applied to the pretrained model, improve performance on intervening months or years (§4.3). The structure can also be used to generalize task-specific models across time periods with analogous time vectors specialized to unlabeled data (§4.4).

Our results show that temporal variation is to some extent encoded in the weight space of finetuned models, and that weight interpolation can help customize language models to new time periods. We publicly release our code, data, and over 500 models finetuned on specific time periods.[1]

2 Data and Finetuning

In this section, we describe our datasets and finetuning techniques, which serve as the basis for all subsequent experiments. We finetune language models on multiple time-stratified datasets, which we use to analyze temporal misalignment and build time vectors. Then, we explore different ways of interpolating between time vectors to generalize to new times. See §4.3-4.5 for more details on interpolation strategies.

2.1 Datasets

Language Modeling We create two new time-specific language modeling datasets from unlabeled text in news and Twitter domains. For these datasets, we measure *perplexity* of the model on the test set:

- WMT Language Modeling: We randomly sample 67K $\pm$ 5K articles (47M BPE tokens) of training articles and 3K $\pm$ 0.3K test articles (2.3–2.4M tokens) from each year 2012–2021 in the English subset of the WMT news dataset (Barrault et al., 2021), from 2012–2016. From the same time range, we also sample 7.1M tokens of training articles and 700–720K tokens of test articles from each month. We are missing WMT train and test splits for August 2012 and May 2016.

- Twitter Language Modeling: We randomly sample 2M $\pm$ 105K training tweets (72–78M tokens BPE tokens) and 100K $\pm$ 5.4K test tweets (3.6-3.9M BPE tokens) from each year in the Internet Archive Twitter Stream Grab,[2] from 2015–2020. We only use this dataset to study the domain-specificity of time vectors in §4.4.

To understand the level of contamination in our datasets, we measure the overlap between yearly train and test splits in both tasks using a Bloom filter.[3] We find that less than two percent and 0.1 percent of examples in the Twitter and WMT LM test sets, respectively, contain contaminated n-grams.

[1] https://github.com/KaiNylund/lm-weights-encode-time

[2] https://archive.org/details/twitterstream

[3] https://github.com/allenai/bff

We do not own any text in these corpora, and publicly release our splits under the CC0 license.

Downstream Tasks For downstream tasks, we draw from Luu et al. (2022). We measure each model's performance on the test set in *ROUGE-L* for NewsSum and *macro F1* for PoliAff.

- NewsSum: We use Luu et al. (2022) postprocessing of Grusky et al. (2018) news summarization task. To align with out WMT dataset, we do not bin adjacent years together, creating uniformly sized splits for each year, 2012–2016.

- PoliAff: We use the Political Affiliation task from Luu et al. (2022), with uniformly sized datasets for each year from 2015 to 2020.

2.2 Finetuning

To compare the same weight space across tasks, we use pretrained T5 (Raffel et al., 2023) checkpoints for all our experiments. We finetune T5-small, T5-large, and T5-3b on each of our time-stratified datsets. For language modeling, we use the "LM adaptation" objective (Lester et al., 2021).

To reduce the computational cost, we finetune T5-large and T5-3B with Low-Rank Adaptation (LoRA; Hu et al., 2021) and default hyperparameters (q and v attention target modules, $r = 8, \alpha = 32$, dropout = 0.1). When creating time vectors, we merge LoRA weights back into the base model before subtracting the pretrained model.

Across all settings, we use a batch size of 2 with 8 gradient accumulation steps. We finetune for a single epoch on LM splits and three epochs on downstream task splits. Our learning rates across all tasks are 8×10^{-4} for T5-small and T5-large, and 2×10^{-4} for T5-3b. We finetuned models concurrently with a single GPU each; we used 8 2080ti, 4 Titan, and 8 A40 GPUs. We use only a single seed for finetuning (42) due to computational constraints. In experiments in §4.4 and §4.5, we ran evaluations in parallel using available Titan, A40, and A100 GPUs.

3 Temporal Misalignment at Multiple Time Scales

We begin with an analysis of temporal misalignment using the new set of models and tasks that we consider in this work (§2). These findings set the stage for our creation of time vectors in §4.

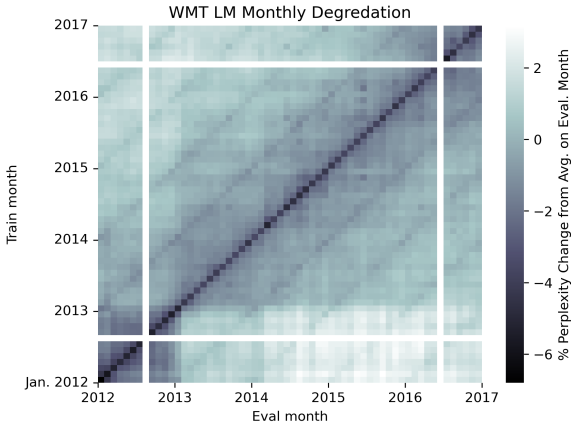

Figure 3: **Monthly temporal degradation has seasonal patterns.** Each cell indicates the monthly performance of T5-small finetuned and evaluated on a *single month* of the WMT dataset. We report the percentage difference in test perplexity from the average on the evaluation month over all finetuned T5-small models (darker is better). The diagonal indicates that each model does best on its finetuning month. Models also do relatively better on the same month in other years, visible as the stripes radiating out from the diagonal every 12 months.

3.1 Yearly Degradation is Linear

Consistent with past work (Lazaridou et al., 2021; Luu et al., 2022; Longpre et al., 2023), we observe linear patterns of year-to-year degradation (Figure 2). We finetune T5-small, T5-large, and T5-3b on each yearly split from every dataset, then evaluate each of these year-finetuned models on every other time split of the test data. Like Luu et al. (2022) show, some tasks, like political affiliation classification, exhibit clearer degradation than others. We quantify these variations in §A.2.

3.2 Monthly Degradation is Seasonal

Next, we turn to month-by-month temporal misalignment. We train T5-small on each WMT LM month split from 2012–2016, resulting in 58 month-finetuned models. We then test every 2012–2016 month model on each month test split for a total of 3,364 evaluations.

Finetuning and evaluating models on specific months in the WMT dataset reveals non-linear patterns in temporal misalignment, which correspond to the cycle of months in each year. This pattern is captured by the stripes that occur parallel to the diagonal every 12 months in Figure 3, which indicate that the model for a particular month tends to do better on the same month in other years. We quantify these differences in perplexity in appendix

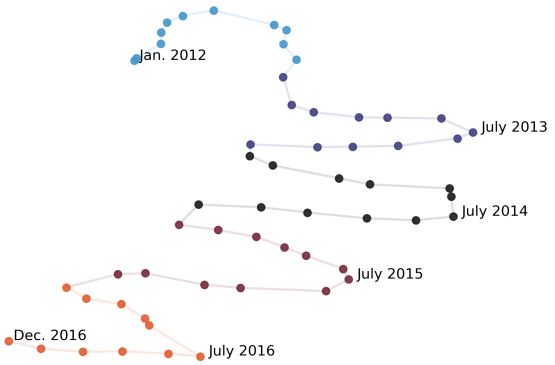

Figure 4: **Time vectors are organized in a manifold that reflects temporal variation.** Each point is a UMAP projection (with default parameters) of the last feedforward layer of a T5-small time vector finetuned on single month of WMT. Points and edges between adjacent months are colored by year. Distances between the weights of time vectors correlate with temporal misalignment (§4.2).

Figure 12. We also report degradation patterns in *online* training settings in §A.4.

3.3 Summary

We measure temporal misalignment across a variety of domains, tasks and time scales. While performance decays linearly on a yearly scale, we discover seasonal trends in month-to-month misalignment. Next, we analyze how these phenomena relate to the weights of time-specific models, and then use that relationship to present techniques for adapting LMs to new times.

4 Temporal Adaptation with Time Vectors

The collection of year and month-finetuned models from §3 presents a new source of data to study temporal misalignment: model weights. In this section, we analyze these weights through the lens of *time vectors*, formed by taking the difference of a model finetuned on a specific time and the pretrained model. First, we show that the weights of two time vectors become less similar as the times they were finetuned on become more misaligned (§4.2). Then, we attempt to use the reverse relationship to update models to unseen times: reducing misalignment on intervening (§4.3), future (§4.4), and multiple time periods (§4.5) by interpolating time vectors.

		Pearson r		
Correlated Metric	T5 size	WMT LM	NewsSum	PoliAff
Time Vec. Similarity	small	-0.867	0.663	0.654
	large	-0.737	0.628	0.672
	3b	-0.795	0.626	0.668
Vocab. Overlap	small	-0.962	0.770	0.887
	large	-0.950	0.758	0.878
	3b	-0.944	0.750	0.862

Table 1: **The similarity between time vectors correlates with temporal degradation, although less than semantic drift.** Pearson correlation between cosine similarity of yearly time vectors, top-10k vocabulary overlap between splits, and % degradation from the mean performance of all yearly models on each evaluation time period. All p-values are $< 9 \times 10^{-4}$.

4.1 Background and Definition

Task vectors (Ilharco et al., 2023) are the difference of the weights of a pretrained model from the weights of the same model after finetuning on a task. Adding and subtracting task vectors from finetuned models is a simple and effective way to improve performance on other settings, or reduce unwanted behavior without further training. Like word embeddings, if there are tasks with the analogous relationship "A is to B as C is to D," then task vectors can be used to improve performance on D with the approximation $D \approx C + (B - A)$.

Time vectors are an extension of task vectors to the time domain. Given the weights of the pretrained model, θ_{pre} and those of the model finetuned on data from only a single time period t, θ_t, a time vector $\tau_t = \theta_t - \theta_{\text{pre}}$. Like their task-based counterparts, we add back the pretrained weights at inference time and evaluate $\theta_{\text{pre}} + \tau_t$ (Ilharco et al., 2023). We call time vectors from models finetuned on individual years and months "year-vectors" and "month-vectors."

4.2 Correlation of Time Vector Similarity and Temporal Degradation

We visualize time vectors with a UMAP in Figure 4, which suggests that time vectors closer together in weight space are also closer together in time. To verify this hypothesis, we measure the cosine similarity between model weights from each pair of time vectors trained on different time periods (visualized in §A.1). As a measure of sematic drift, we also calculate the percentage overlap between the top-10k most frequent white-space separated tokens in each train and test split.

We find that time vector similarity and performance (Figure 11) decay similarly over time. Table

1 shows that the correlation between cosine similarity and relative performance change on different years is highest in WMT language modeling. Correlations are generally similar across T5 sizes, with a higher score for T5-small in the WMT LM setting than T5-large and T5-3b, and no absolute values less than 0.6.

Mirroring these findings, vocabulary overlap between splits is an even better predictor of degradation between misaligned times, with correlations consistently 0.1–0.2 higher than time vector similarity. Because explicit dates (e.g. "2014", "december", "7/2/2013") likely make up a miniscule percentage of the top-10k tokens in each split, we expect that semantic shift has a larger impact on temporal degradation.

These correlations extend to the monthly scale. Seasonal stripes are visible in the cosine similarities between each pair of monthly WMT time vectors (visualized in Appendix Figure 9). The monthly performance degradation from the mean (Figure 3) has a negative correlation with both month vector similarity (Pearson $r = -0.667$; $p < 10^{-16}$) and month-to-month vocabulary overlap (Pearson $r = -0.886$; $p < 10^{-16}$). We analyze cosine similarities to single-year time vectors throughout online training in Appendix §A.5.

These results indicate that time vectors are organized in way that is predictive of their performance on corresponding time periods. Next, we explore how we can use this structure to improve on new time periods by interpolating between time vectors.

4.3 Generalizing to Intervening Time Periods

Archiving issues or a low sampling rate can lead to gaps in datasets between the oldest and newest examples. Without data, we expect models to perform worse on these "gap" times due to temporal misalignment. In this section, we find that we can generalize better to these intervening time periods by mixing models finetuned on the oldest and newest times with intuitive coefficients.

Method For two time vectors τ_j, τ_k, we compute their interpolation $\alpha \cdot \tau_j + (1 - \alpha) \cdot \tau_k$ with $\alpha \in [0, 1]$. In this section, we interpolate between the earliest year time vector τ_0 and latest year time vector τ_n and evaluate on times $t_0, ..., t_n$ for each $\alpha \in [0.1, 0.2, ..., 1.0]$.

Results Figure 5 shows that interpolating between start and end-year finetuned models improves performance on intervening years in both WMT LM and PoliAff tasks. Improvement is generally greatest on the exact middle years (2014 for WMT LM, 2017 for PoliAff) and decreases on years closer to start and end times. Patterns of improvement also vary depending on setting, with flatter changes in performance near $\alpha = 1.0, 0.0$ in PoliAff compared to WMT LM, and minimal improvements in NewsSum across αs compared to the difference in performance between evaluation years. Table 2 quantifies these changes, showing that interpolation closes the gap on intervening years between temporally aligned and misaligned models. Improvements are particularly large for PoliAff, nearly eight macro-F1 points just by averaging the start and end-year time vectors.

	Perplexity (↓)	*Rouge* (↑)	*F1* (↑)
Method	WMT LM	NewsSum	PoliAff
Start-year finetuned (τ_0)	13.92	38.56	0.6886
End-year finetuned (τ_n)	13.84	35.09	0.6967
$\frac{1}{2}(\tau_0 + \tau_n)$	13.77	38.86	0.7765
Best interpolations	13.75	40.11	0.7941
Eval-year finetuned (τ_i)	**13.65**	**42.36**	**0.8341**

Table 2: **Interpolation between start and end-year finetuned models reduces temporal misalignment on intervening years.** T5-3b average performance on each year between start and end (non-inclusive). "Best interpolations" use the best performing α values for each year.

Figure 6 shows that these results extend to the monthly scale for WMT LM; we can interpolate between time vectors finetuned on January and December in a year to improve performance on the months between them. The best interpolations for each month follow an intuitive pattern, with a higher percentage of the January model leading to better performance on earlier months and vice versa.

4.4 Generalizing to the Future

The creation of labeled datasets lags behind corpera of raw text, which can be scraped automatically. As a result, language models that rely on supervision for finetuning are quickly outdated. Updating these models can be expensive, involving extra finetuning and creating labeled datasets from more recent examples. In this section, we present a new technique for updating task models finetuned on a source time period j to a target time period k with only unlabeled data from j, using task analogies (Ilharco et al., 2023).

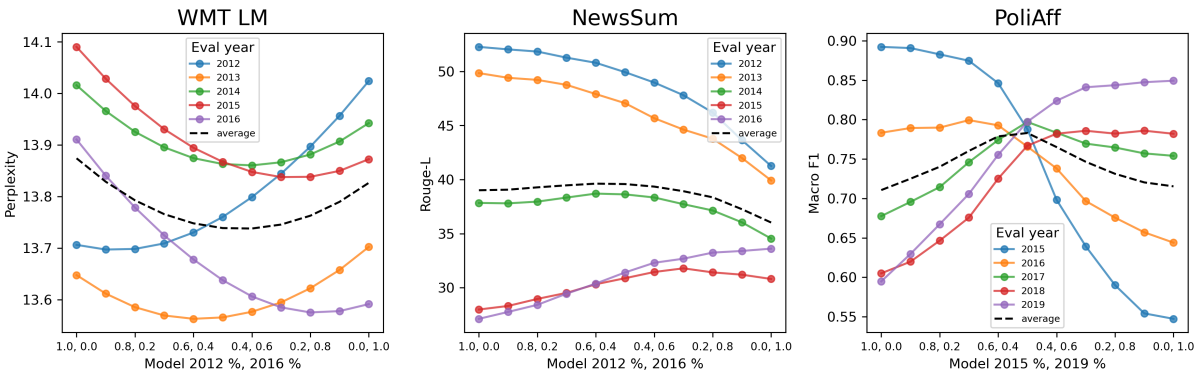

Figure 5: **Interpolating between two year vectors improves performance on the years between them.** T5-3b performance improvements follow an intuitive structure, e.g. when interpolating between 2012 and 2016, the best result on 2013 occurs with a higher percentage of 2012 and vice versa for 2015. Improvement from interpolation varies across settings.

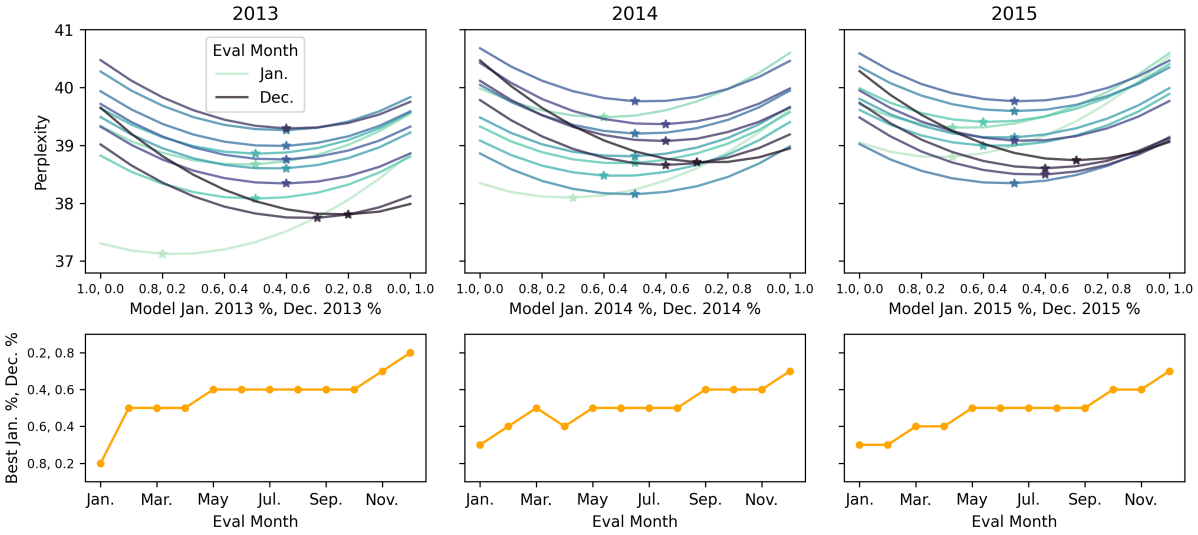

Figure 6: **Interpolating between two month vectors improves performance on the months between them.** We interpolate between WMT January and December month vectors and evaluate on all other months within the same finetuning year. Like at the yearly scale, early months do better with a higher percentage of the January model and vice versa while middle months do best with a 50% split between the models. The lower row of plots show the best alpha value for each evaluation month, represented with stars in the top row.

Method Given language models with weights $\theta_j^{\text{LM}}, \theta_k^{\text{LM}}$ finetuned on unlabeled text from times j, k, and a task-specific model with weights θ_j finetuned on labeled data from time j, we perform the following arithmetic on the vectors:

$$\begin{aligned}
\tau_j &= \theta_j - \theta_{pre} \\
\tau_j^{\text{LM}} &= \theta_j^{\text{LM}} - \theta_{pre} \\
\tau_k^{\text{LM}} &= \theta_k^{\text{LM}} - \theta_{pre} \\
\tau_k &\approx \alpha_1 \cdot \tau_j + (\alpha_2 \cdot \tau_k^{\text{LM}} - \alpha_3 \cdot \tau_j^{\text{LM}}) \\
\theta_k &= \tau_k + \theta_{pre}
\end{aligned}$$

We evaluate our estimated θ_k on each target time t_k, sweeping over all combinations of $\alpha_1 \in [0.6, 0.8, \ldots 2.2], \alpha_2, \alpha_3 \in [0.1, \ldots 0.6]$ and reporting the best result compared to the original model θ_j. In this section, we update a 2012 NewsSum model to 2013–2016, and a 2015 PoliAff model to 2016–2020 using WMT LM and Twitter LM time vectors respectively.

Results Task analogies improve performance on future years in both PoliAff and NewsSum tasks. Figure 7 shows that improvement compared to finetuning on the start year increases as the target and start years become more misaligned. Model size also affects performance, with T5-large and T5-3b showing greater improvements. In PoliAff, T5-small has no improvement over the baseline and

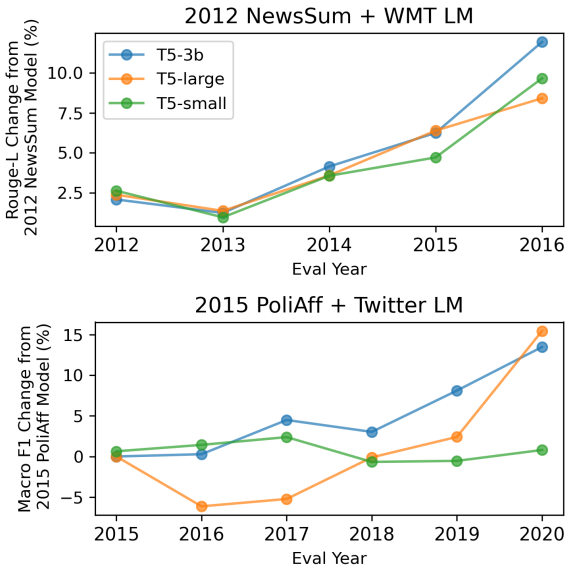

Figure 7: **Task analogies can offset downstream temporal misalignment without labeled data from the target time.** We report the performance of NewsSum and PoliAff T5 models updated using WMT LM and Twitter LM vectors for each target evaluation time. We report the percent improvement of the best updated model over 2012 NewsSum and 2015 PoliAff models on each target time for all model sizes.

T5-large task analogies perform worse than the baseline on 2016 and 2017 before improving on 2019 and 2020. We observe mostly similar results on these tasks, although there are task-specific inconsistencies.

Strangely, we find that only scaling α_1 can also improve performance on future years. This phenomenon could be a proxy for up or down-weighting the data of the source time. When the pretraining data is closer to the target time than the source data, for instance, we can improve solely by down-weighting the source time vector by choosing $\alpha_1 < 1.0$ and vice versa. We report ablations on α values and our results on two other classification tasks in Appendix §A.6.

4.5 Generalizing to Multiple Time Periods

Because interpolations prove useful for generalizing to intervening and future time periods, we next test if we can build models that perform well on *multiple* time periods by interpolating between all time vectors for a task.

Method We approach this problem with the *model soup* technique (Wortsman et al., 2022). One of the key practical advantages of soups is that constituent time-specific models can be trained independently (on smaller compute budgets) and combined at any time. Furthermore, the multi-year model does not need to be retrained to include new time periods; new time periods can be incorporated by merely growing the soup with additional finetuned models.

We attempt to create a multi-year model by following the recipe outlined by Wortsman et al. (2022). They introduce two soup variants: the *uniform soup* and *greedy soup*. The uniform soup applies a uniform weight among all constituent models in the interpolation, while the greedy soup is an iterative procedure that only includes models in the soup that improves validation performance. We assess both variants here.

Our "uniform time soup" is $\theta_{\text{pre}} + \frac{1}{|T|}\sum_{t\in T}\tau_t$ where T is the set of all years for a given task. For our "greedy time soup," we implement a similar algorithm to Wortsman et al. (2022) which samples time vectors (with replacement) from each year in order of decreasing performance and adds them to the average model soup if they improve performance.

To evaluate our ability to build models that generalize to multiple time periods, we measure the average performance across all evaluation years for each task. We compare our model soups against two baselines: 1) a model trained on all shuffled available data at once and 2) the best-performing model finetuned on only a single year of data. The all-year model is the most compute-intensive approach.

Results Overwhelmingly, time soups perform worse than the model finetuned on all shuffled available data. For WMT LM and NewsSum, the uniform time soup performs worse than even the best single year model, despite having access to five times the amount of finetuning data. The greedy time soup only improves over the best single-year model on PoliAff with a single macro F1 point gain. These findings suggest that a model which generalizes to multiple time periods does not lie in a region of weight space bounded by models finetuned on single years of data. Future work may explore more sophisticated methods of merging to induce better performing multi-year models.

4.6 Summary

We propose methods for updating models to intervening, future, and multiple time periods using time vector arithmetic. We find that interpolating

	Perplexity (↓)	*Rouge* (↑)	*F1* (↑)
Method	WMT LM	NewsSum	PoliAff
Best single-year model	34.45	38.95	0.7101
Uniform time soup	34.70	33.05	0.6078
Greedy time soup	34.45	38.95	0.7202
Training on all years	**29.17**	**40.07**	**0.7853**

Table 3: **Interpolation does not enable generalization to multiple time periods simultaneously.** Here, we measure the average performance of models on all years. We compare multiple ways of building multi-year models; T5-small models finetuned to individual years or all years, and "time soups" created by averaging together all year time vectors for a task.

between two time vectors improves performance on unseen intervening times at both yearly and monthly scales. Similarly, we can improve performance on the future with unlabeled data from target times using time vector analogies. Building a multi-year model with a "soup" of time vectors, however, does not approach the performance of a model finetuned on all times at once. These results suggest that task arithmetic can be a simple way to update models to new times, but it does not help to improve generalization across the board within a single model.

5 Related Work

Semantic Drift Although changes in the full weight spaces of models over time have not been previously explored, semantic changes in word embeddings over time are well-documented (Hamilton et al., 2016). Temporal misalignment (Bamler and Mandt, 2017; Gonen et al., 2021) and word analogies over time (Szymanski, 2017) have also been studied in embeddings. Our work extends these analyses to the full set of language model parameters.

Temporal Misalignment The phenomenon of temporal misalignment in language models has gained attention in the last three years. Moving from semantic drift to model misalignment, temporal degradation has been studied in a variety of tasks including gender and age classification (Jaidka et al., 2018), named entity recognition (Rijhwani and Preoţiuc-Pietro, 2020; Liu and Ritter, 2022), summarization (Cheang et al., 2023), language modeling (Loureiro et al., 2022), and many others (Yao et al., 2022). Lazaridou et al. (2021) additionally show that increasing model size does not help mitigate temporal misalignment, and Luu et al. (2022) find that degradation varies greatly over both domain and task. Longpre et al. (2023) report similar decay over time in pretraining.

Updating LMs Recent attempts at updating language models to new time periods have used a range of techniques. Röttger and Pierrehumbert (2021) and Luu et al. (2022) find limited downstream improvement with continued pretraining on target times. Yao et al. (2022) find similar negative results with invariant, continual, and ensemble learning approaches on their dataset of in-the-wild downstream tasks. Similar to the sequential updating setting, however, Lazaridou et al. (2021) show that dynamic evaluation (Gururangan et al., 2020) can improve language modeling performance on new times, but results in forgetting the past. Other techniques have been proposed for keeping models up to date in the QA domain by adding flags with the year for each example (Dhingra et al., 2022) or by discarding outdated facts (Zhang and Choi, 2023). Similarly, Su et al. (2022) improve on language modeling and classification tasks by masking out tokens subject to semantic drift during finetuning. Unlike these methods, we consider the problem of updating models to new time periods without data in the target time and without additional training.

Interpolation Our work draws heavily on recent techniques for editing models directly with interpolation and task analogies. Time vectors are an application of task vectors (Ilharco et al., 2023) to the time domain, our interpolation experiments are inspired by previous work on patching models for multiple tasks (Ilharco et al., 2022), and our time soups are an application of models soups (averaging multiple models trained with different initializations; Wortsman et al., 2022).

6 Conclusion

We connect studies of temporal misalignment and weight arithmetic with time vectors, formed by finetuning a model on a specific time period and then subtracting its pretrained weights. We show that the weights of time vectors are more similar if their corresponding times are closer and vice versa. These similarities are highly correlated to temporal misalignment at both yearly and monthly scales (which exhibit seasonal patterns). Leveraging this temporal structure in weight space, we induce new models that perform better on intervening years by

interpolating between adjacent time vectors. Similarly, we use task analogies to improve downstream performance on future time periods using only unlabeled data from those times. These results show that task arithmetic can be a simple tool for updating models to new time periods.

7 Limitations

Our analyses are restricted to three sizes of T5, with the largest containing three billion parameters. Because we use LoRA when finetuning T5-large and T5-3b, our total number of trainable parameters never exceeded those of base T5-small ($\sim$60 million). Although similar patterns of temporal misalignment have been observed in larger autoregressive models (e.g., Luu et al., 2022, Longpre et al., 2023), improvements from time vector arithmetic are not guaranteed to scale with mutli-billion parameter LMs.

Because we only finetune and evaluate at the monthly scale with WMT news articles, seasonality may not occur, or occur differently, in other domains. News may be particularly suited to seasonal trends in perplexity compared to, e.g., fiction novels, because of reporting on events like holidays and weather.

In time vector analogies, unlabeled text may still be difficult to gather for isolated source and target times due to a lack of metadata. Furthermore, finding ideal α values for each vector in the analogy arithmetic requires searching over a large number of combinations (324 in our experiments). In the best case, we find that time vector analogies can improve performance 5–15% on a target year over a model finetuned on only the earliest year. These improvements vary by task, however, and analogies can even hurt performance in some cases, as we show in Appendix §A.6.

In practice, models are trained on text from many time periods at once, which likely yields better results than a single time-specific model. Our experiments with time vectors are therefore focused on analyzing the relationship between time-specific models in weight space, and the potential of weight arithmetic for adapting models to new times, rather than improving the state of the art.

8 Ethical Considerations

For further study of temporal misalignment and replication of our experiments, we publicly release our models finetuned on text from the monolingual WMT news dataset and Twitter stream grab. Following guidelines from both sources, all models are under a CC0 license and should be used solely for research.

Corpora and tasks used in this dataset do not identify authors of examples, but include information about other individuals, including which user a post is retweeting in the Twitter splits. Although frequently mentioned names are important features for studying temporal variation, we realize that our models may reproduce this identifying information in their outputs alongside falsehoods or hallucinations. Because we do not filter out examples containing toxic or offensive language in our datasets, we acknowledge that the models we release are susceptible to generating text which perpetuates social harms (Gehman et al., 2020).

Although we aim to cover a range of downstream tasks for each year and monthly domain shift, our datasets are not equally representative of different languages and demographic groups. We filter out documents that are not classified as English, and note that the majority of news articles and tweets are sourced from the U.S., where the majority of journalists are white and between the ages of 30–64 (Tomasik and Gottfried, 2023). As a result, our models finetuned on NewsSum and WMT are likely harmfully skewed towards white-aligned English, reproducing the view that other registers are linguistically inadequate (Rosa and Flores, 2017).

Finally, we are cognizant that finetuning year and month-specific models incurs a significant energy cost. We estimate that it took on average three hours to train each T5-small and T5-large model on yearly WMT splits, and nine hours for T5-3b. Training on NewsSum splits took around a third of the time as WMT LM. For PoliAff, the train time for year-finetuned models was lower at around 5 minutes for T5-small and T5-large, and 15 minutes for T5-3b. Finetuning T5-small on a single monthly WMT split took 15 minutes on average. Evaluating on each split took roughly a tenth of the time as training. Using these heuristics, we estimate the main paper experiments took a total of 1200 GPU hours. We did not track GPU usage on preliminary or Appendix experiments, but we estimate they used an equivalent 1200 GPU hours.

9 Acknowledgments

We thank Gabriel Ilharco and the anonymous reviewers for feedback on drafts.

References

Robert Bamler and Stephan Mandt. 2017. Dynamic word embeddings. In *International conference on Machine learning*, pages 380–389. PMLR.

Loic Barrault, Ondrej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussa, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Tom Kocmi, Andre Martins, Makoto Morishita, and Christof Monz, editors. 2021. *Proceedings of the Sixth Conference on Machine Translation*. Association for Computational Linguistics, Online.

Chi Cheang, Hou Chan, Derek Wong, Xuebo Liu, Zhaocong Li, Yanming Sun, Shudong Liu, and Lidia Chao. 2023. Can lms generalize to future data? an empirical analysis on text summarization. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16205–16217.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Hila Gonen, Ganesh Jawahar, Djamé Seddah, and Yoav Goldberg. 2021. Simple, interpretable and stable method for detecting words with usage change across corpora. *arXiv preprint arXiv:2112.14330*.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *arXiv preprint arXiv:1804.11283*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. *ArXiv*, abs/2004.10964.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*.

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.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. 2023. Editing models with task arithmetic.

Gabriel Ilharco, Mitchell Wortsman, Samir Yitzhak Gadre, Shuran Song, Hannaneh Hajishirzi, Simon Kornblith, Ali Farhadi, and Ludwig Schmidt. 2022. Patching open-vocabulary models by interpolating weights. *Advances in Neural Information Processing Systems*, 35:29262–29277.

Kokil Jaidka, Niyati Chhaya, and Lyle Ungar. 2018. Diachronic degradation of language models: Insights from social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 195–200.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning.

Margaret Li, Suchin Gururangan, Tim Dettmers, Mike Lewis, Tim Althoff, Noah A. Smith, and Luke Zettlemoyer. 2022. Branch-train-merge: Embarrassingly parallel training of expert language models.

Shuheng Liu and Alan Ritter. 2022. Do conll-2003 named entity taggers still work well in 2023? *arXiv preprint arXiv:2212.09747*.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, et al. 2023. A pretrainer's guide to training data: Measuring the effects of data age, domain coverage, quality, & toxicity. *arXiv preprint arXiv:2305.13169*.

Daniel Loureiro, Francesco Barbieri, Leonardo Neves, Luis Espinosa Anke, and Jose Camacho-Collados. 2022. Timelms: Diachronic language models from twitter. *arXiv preprint arXiv:2202.03829*.

Kelvin Luu, Daniel Khashabi, Suchin Gururangan, Karishma Mandyam, and Noah A. Smith. 2022. Time waits for no one! analysis and challenges of temporal misalignment. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5944–5958, Seattle, United States. Association for Computational Linguistics.

Guillermo Ortiz-Jiménez, Alessandro Favero, and Pascal Frossard. 2023. Task arithmetic in the tangent space: Improved editing of pre-trained models. *ArXiv*, abs/2305.12827.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer.

Shruti Rijhwani and Daniel Preoţiuc-Pietro. 2020. Temporally-informed analysis of named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7605–7617.

Jonathan Rosa and Nelson Flores. 2017. Unsettling race and language: Toward a raciolinguistic perspective. *Language in society*, 46(5):621–647.

Paul Röttger and Janet B Pierrehumbert. 2021. Temporal adaptation of bert and performance on downstream document classification: Insights from social media. *arXiv preprint arXiv:2104.08116*.

Zhaochen Su, Zecheng Tang, Xinyan Guan, Juntao Li, Lijun Wu, and Min Zhang. 2022. Improving temporal generalization of pre-trained language models with lexical semantic change. *arXiv preprint arXiv:2210.17127*.

Terrence Szymanski. 2017. Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In *Proceedings of the 55th annual meeting of the association for computational linguistics (volume 2: short papers)*, pages 448–453.

Emily Tomasik and Jeffrey Gottfried. 2023. Us journalists' beats vary widely by gender and other factors.

Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, et al. 2022. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pages 23965–23998. PMLR.

Mitchell Wortsman, Gabriel Ilharco, Mike Li, Jong Wook Kim, Hannaneh Hajishirzi, Ali Farhadi, Hongseok Namkoong, and Ludwig Schmidt. 2021. Robust fine-tuning of zero-shot models. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7949–7961.

Huaxiu Yao, Caroline Choi, Bochuan Cao, Yoonho Lee, Pang Wei W Koh, and Chelsea Finn. 2022. Wild-time: A benchmark of in-the-wild distribution shift over time. *Advances in Neural Information Processing Systems*, 35:10309–10324.

Michael JQ Zhang and Eunsol Choi. 2023. Mitigating temporal misalignment by discarding outdated facts. *arXiv preprint arXiv:2305.14824*.

A Appendix

A.1 Yearly Misalignment with Other Tasks and T5 Sizes

In this section, we report raw performance degradation over time on four downstream and three language modeling tasks with three sizes of T5. We evaluate on all tasks in the main paper plus Newsroom Source Classification (NewsCls) and AI Venue Classification (AIC) from Luu et al. (2022). We also create a third science domain language modeling task from abstracts in the Kaggle arXiv dataset[4]. For each group of three years from 2006-2008 to 2018-2020 we randomly sample 26-38M and 2.6-3.9M BPE tokens (150MB and 15MB) of arXiv paper abstracts for train and test splits respectively.

Figures 8 and 11 are yearly degradation heatmaps for each model size and task. These results show that normalizing performance by the average on each evaluation time helps account for variations in test splits. ArXiv language modeling and NewsSum, for example, have large differences in performance on evaluation years regardless of finetuning year.

A.2 Task Variations in Linear Yearly Degradation

Like Luu et al. (2022), we find differences across domain and task in the rate and linearity of year-to-year decay. TD scores measure the average rate of performance degredation for each year of misalignment between train and test time periods (Luu et al., 2022). We find the rate of decay using a linear least squares regression and average rates for each task over all evaluations. Table 4 shows TD scores (Luu et al., 2022) for all tasks and T5-sizes. We also compare TD scores calculated from raw performance to TD scores calculated from performance normalized by the average on each evaluation year. In general, percent performance difference from the mean on an evaluation year decays more linearly than raw performance.

A.3 Yearly and Monthly Cosine Similarities

In this section, we report cosine similarity between each pair of yearly and monthly time vectors. Figure 10 shows cosine similarity between every pair of year vectors for each T5-size and task. Figure 9

[4] https://www.kaggle.com/datasets/Cornell-University/arxiv/data

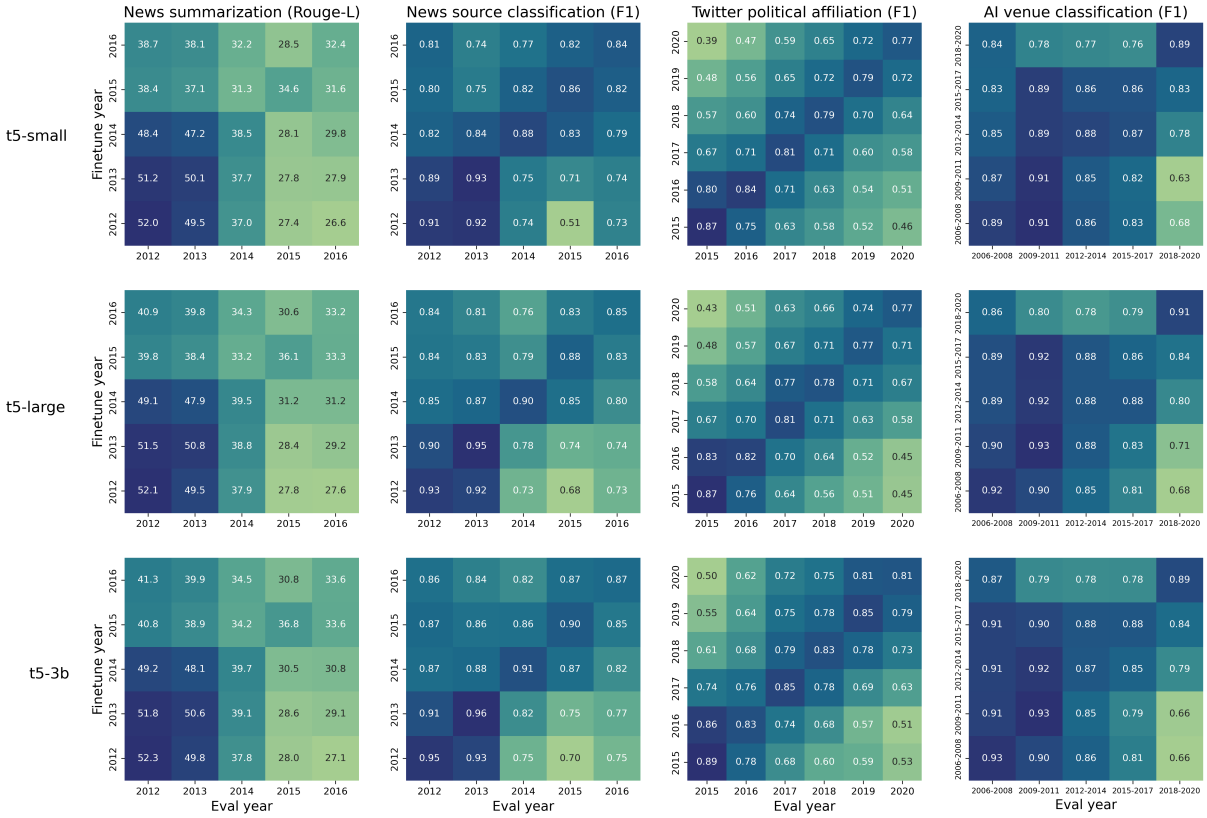

Figure 8: Yearly downstream performance degradation on four tasks and three T5 sizes.

Normalized?	T5 Size	WMT LM	NewsSum	NewsCls	Twitter LM	PoliAff	ArXiv LM	AIC
No	small	-0.67 (0.81)	2.21 (0.51)	0.05 (0.67)	-0.35 (0.97)	0.08 (0.98)	-0.59 (0.65)	0.03 (0.55)
	large	-0.10 (0.34)	2.07 (0.53)	0.04 (0.61)	-0.20 (0.97)	0.07 (0.97)	-0.20 (0.67)	0.03 (0.50)
	3b	-0.07 (0.34)	2.12 (0.53)	0.04 (0.67)	-0.20 (0.97)	0.07 (0.95)	-0.13 (0.66)	0.03 (0.40)
Yes	small	-1.70 (0.90)	6.99 (0.87)	6.43 (0.74)	-4.52 (0.89)	10.47 (0.95)	-2.61 (0.94)	2.93 (0.57)
	large	-0.56 (0.92)	6.27 (0.89)	5.33 (0.84)	-2.64 (0.91)	9.57 (0.94)	-1.24 (0.93)	2.53 (0.51)
	3b	-0.52 (0.93)	6.44 (0.88)	4.72 (0.84)	-2.90 (0.91)	7.66 (0.91)	-0.96 (0.94)	3.12 (0.61)

Table 4: TD scores for all tasks and T5 sizes for raw performance and performance divide by the average on each eval. year. Variance explained by the TD score linear fit in parentheses. TD scores calculated with normalized performance decay have generally higher R^2 scores, except on Twitter LM and PoliAff, and are easier to compare.

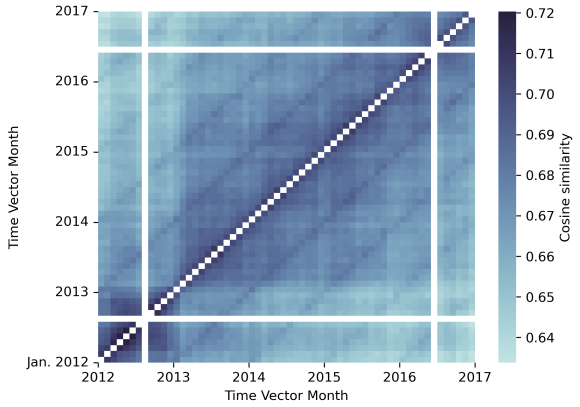

Figure 9: **Cosine similarity between monthly time vectors also exhibits seasonality.** We observe similar "stripes" every 12 months when measuring the cosine similarity between each pair of T5-small WMT month vectors. The correlation between this heatmap (including the diagonal) and Figure 3 is -0.667 with $p < 1 \times 10^{-16}$.

shows cosine similarity between each pair of T5-small monthly WMT LM time vectors. Similar to performance, year-to-year degradation in cosine similarity between task vectors appears to be linear regardless of setting. Like Figure 3, we observe seasonal "stripes" every 12 months from the diagonal 9.

A.4 Temporal Degradation in Online Settings

Our work so far illustrates temporal misalignment on static time splits. However, in practice, we usually deploy language models in online settings, meaning that they are continually updated with the latest data, and we do not have access to data from all training years simultaneously.

To show how temporal misalignment manifests in these settings, we first sort all the training data from the PoliAff and WMT tasks by month, and finetune T5-small on each task separately. We display the performance of the LM on every year throughout training in Figure 13. As expected, for PoliAff, we see that the performance of models on a particular year peak at the final month of that year, and then gradually degrade as the model continues training.

For language modeling on WMT data, performance consistently *improves* during training, regardless of the evaluation year. However, perplexity reduces more slowly in earlier years as we continue training. These results suggest that temporal misalignment may manifest differently in online settings based on the training setup and task.

A.5 Online Cosine Similarities

We study the relationship between performance degradation and cosine similarity during *online training*. Recall that in the online setting, we perform a single finetuning run on the Poliaff and WMT tasks (after ordering their training data by month), and measure performance on each year throughout training. To study how time vectors move throughout space in this setting, we measure the cosine similarity between the time vector of the model trained up to month m and each yearly time vector for the PoliAff and WMT tasks.

We find that the cosine similarity to each time vector *decreases* as the online model is updated past the first 12 months of data. This means that online models' peak similarity to earlier years tends to be higher than those to later years since the they make up a smaller part of its total finetuning set. Like our experiments with soups of time vectors in section §4.5, this indicates that models trained on multiple years of data lay outside a region defined by single-year models.

To account for these decreases, we normalize the similarity to each year time vector by its average after updating on all months in Figure 13. Our results reveal that the vector for our online model is relatively most similar to each year vector after finetuning on the months in that year.

A.6 Time Vector Analogy Ablations

In this section, we ablate our time vector analogy experiment to determine the effects of only adding the LM vector from the target time, and only scaling the weights of the initial time vector. For $\tau_k \approx \alpha_1 \cdot \tau_j + (\alpha_2 \cdot \tau_k^{\text{LM}} - \alpha_3 \cdot \tau_j^{\text{LM}})$, we define our "task addition" ablation for $\alpha_3 = 0, \alpha_1, \alpha_2 \neq 0$, and our "scaling only" ablation for $\alpha_1 \neq 0, \alpha_2, \alpha_3 = 0$

We report the best results after sweeping over the same α ranges from §4.4 with the added constraints in figure 15. While task analogies generally perform best across tasks and T5-sizes (especially as τ_j and τ_k become more misaligned), we find that ablating τ_k^{LM} and τ_j^{LM} can still improve over the base τ_j model. Surprisingly, *only scaling* τ_j also improves over the initial model on many tasks.

A.7 Temporal Misalignment Affects Some Parameters More than Others

In this section, we explore whether we can reduce temporal misalignment by swapping parameter weights from a model trained on a misaligned

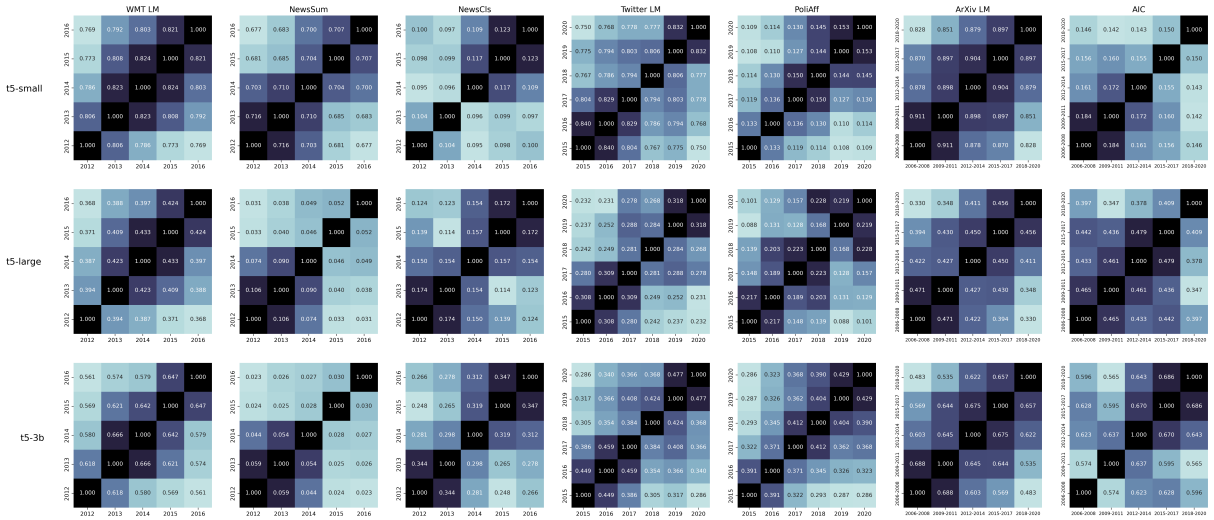

Figure 10: Cosine similarities between all pairs of year time vectors for all tasks and model sizes.

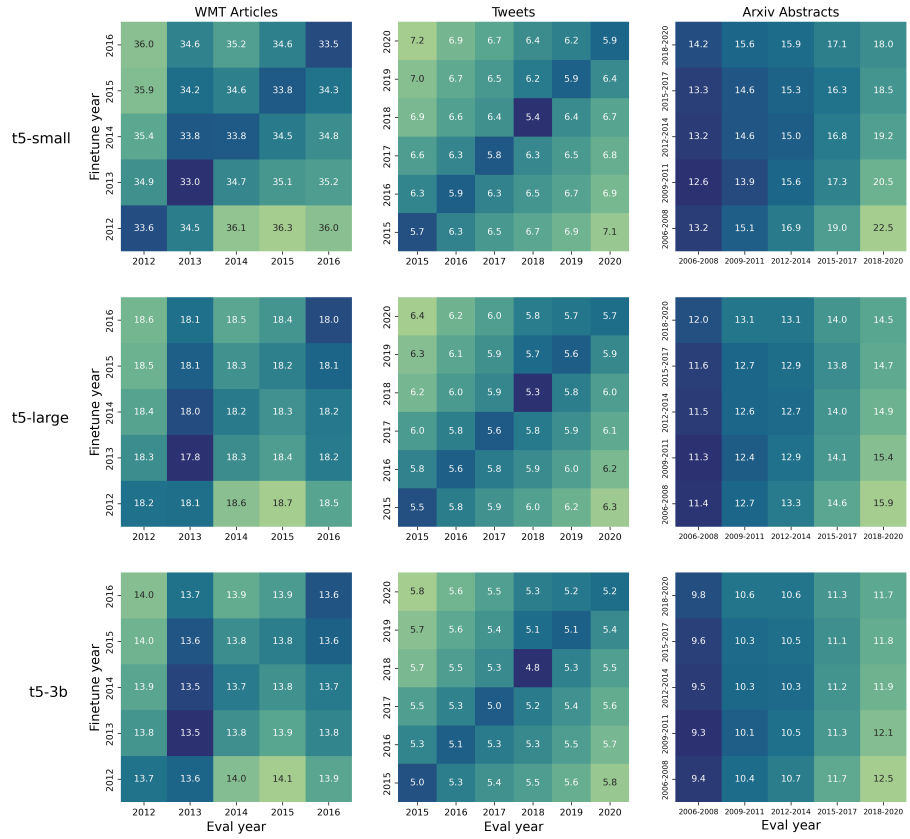

Figure 11: Yearly language modeling perplexity decay on three tasks and three T5 sizes.

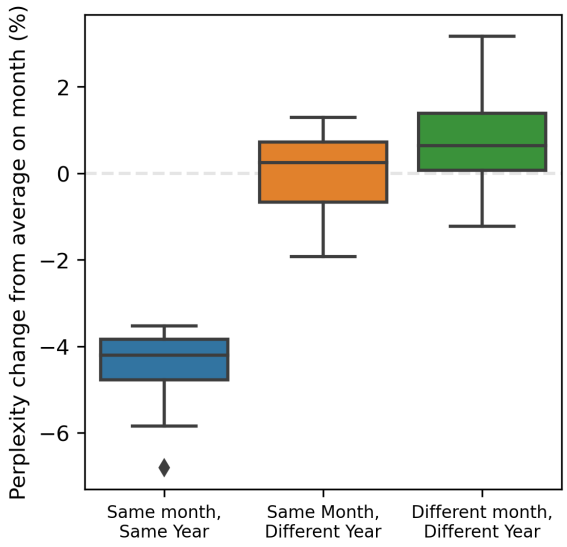

Figure 12: **Seasonality makes a small, but noticeable impact on monthly misalignment.** Distribution of perplexity change from the mean for aligned finetuning and evaluation months (left, mean=-4.36), seasonal "stripes" (middle, mean=0.04), and all finetuning and evaluation combinations which share neither the same month nor year (right, mean=0.77).

year with those of the model trained on the target year. For example, we substitute the QKV attention layers from a model finetuned on 2015 PoliAff with those finetuned on 2020 PoliAff and evaluate on 2020 data. In table 5 we evaluate the start-year finetuned models for each task on the end times (e.g. start = 2012 for WMT LM, end = 2016) with various parameter weights swapped with the end-year finetuned model.

From these experiments, we find that we can improve performance on a target time by swapping out weights with a time vector finetuned on that time. Surprisingly, swapping embeddings with the target time vector makes very little difference, except in language modeling tasks, and swapping all non-embedding weights with a target time almost reaches the performance the target time-specific models for downstream tasks. Swapping only feed-forward or attention layers also improves performance on the target time, suggesting temporal misalignment is somewhat isolated to those model regions in downstream tasks.

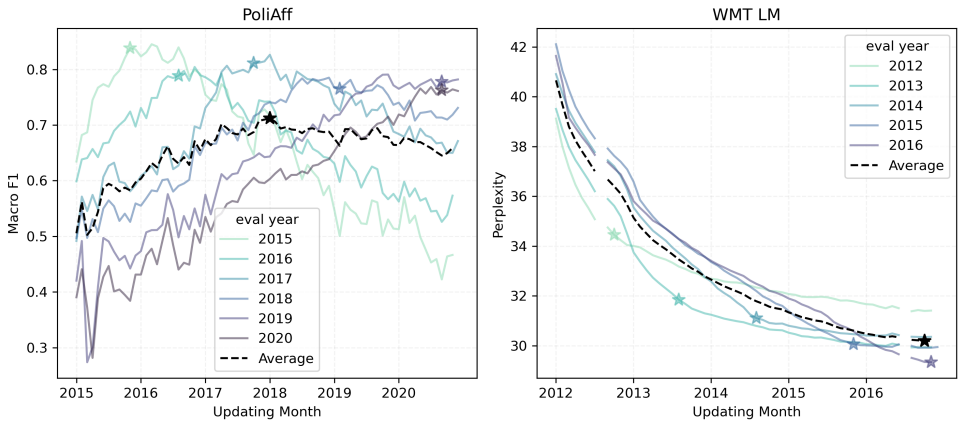

Figure 13: **In online settings, language model performance degrades on earlier time periods.** We show macro F1 and perplexity on each year split of PoliAff and WMT LM respectively after sequentially finetuning T5-small on each new month of task data. PoliAff performance over all years plateaus after finetuning on months up to 2018. WMT performance continues to improve with more data, but perplexity decrease slows on earlier years. Starred points are where performance on a year is best relative to the average performance on all years.

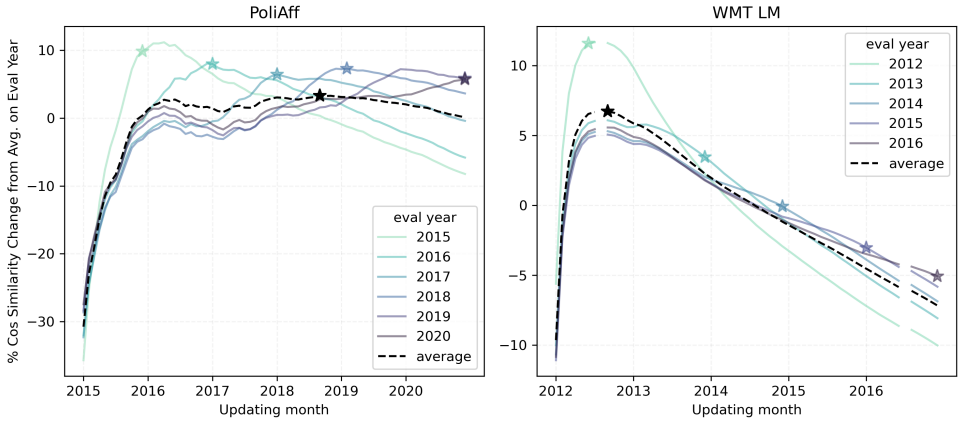

Figure 14: **Cosine similarity between an online time vector and a year vector peaks relative to other years after updating on data for that year.** We show cosine similarity between each monthly checkpoint of online T5-small time vectors and yearly vectors for PoliAff and WMT LM. To account for overall decreases in similarity as online time vectors are updated, we normalize similarities to each year vector by the mean similarity to that year over all checkpoints. We star the point for each year vector where its cosine similarity to the online model is largest relative to the average on all years.

Swapped Params	WMT LM	NewsSum	NewsCls	Twitter LM	PoliAff	ArXiv LM	AIC
None	*35.72*	*35.11*	*0.7232*	*6.69*	*0.5903*	*18.18*	*0.8224*
Feed Forward	35.31	35.17	0.8162	**13.25**	0.6174	18.21	0.8500
Attention	36.23	34.49	0.7986	14.95	0.6095	19.24	0.8644
Embeddings	36.13	34.30	0.7232	16.65	0.5902	19.29	0.8192
Non-Embedding	**34.57**	**37.24**	**0.8760**	**13.46**	**0.7991**	**17.37**	**0.8845**
All	*33.51*	*38.89*	*0.8759*	*5.79*	*0.7999*	*15.75*	*0.8845*

Table 5: **We can improve performance on a target time by swapping out weights with a time vector finetuned on that time.** T5-small start-year finetuned model performance on the end-year split for each task (e.g. finetuning on 2015 for PoliAff and evaluating on 2020). We compare the baseline start-year model (none swapped) to versions with various parameter weights from the target-year model, and the target-year model itself (all swapped).

Figure 15: Time vector analogy ablations for three sizes of T5. Given the time vector analogy $\tau_k \approx \alpha_1 \cdot \tau_j + (\alpha_2 \cdot \tau_k^{\text{LM}} - \alpha_3 \cdot \tau_j^{\text{LM}}), \alpha_1, \alpha_2, \alpha_3 \neq 0$, we define "task addition" to be only adding the language modeling vector (i.e $\alpha_1, \alpha_2 \neq 0, \alpha_3 = 0$), and "scaling only" to be only scaling the base τ_j model (i.e $\alpha_1 \neq 0, \alpha_2, \alpha_3 = 0$). We sweep over the same α combinations as in §4.4 and report the best results for each target year, task, and T5-size.

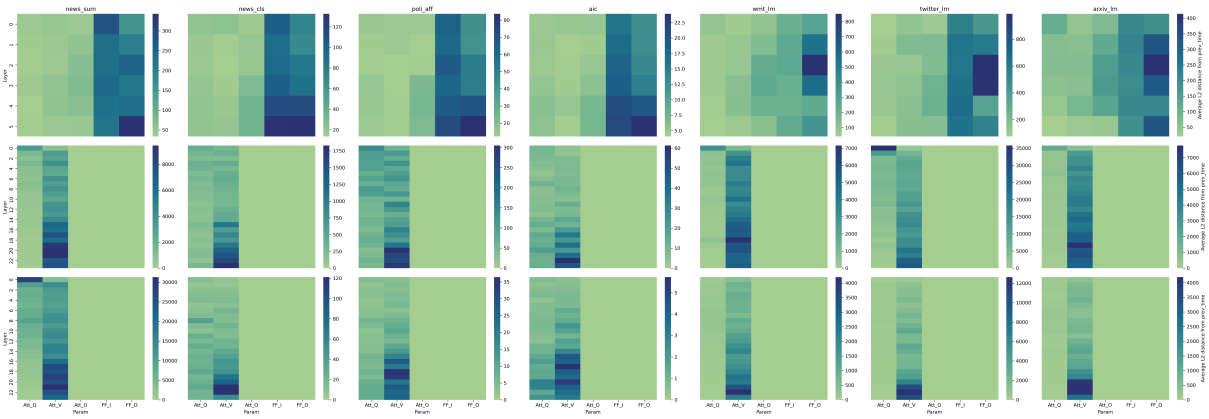

Figure 16: Year-to-year, T5-small feed forward layers change the most across all tasks and domains, and attention changes more in the language modeling setting. For our T5-large and T5-3b models trained with LoRA, the V attention layers change more than the Q layers, with most of the changes (regardless of model size) concentrated in the last layers. Like our param swapping experiment, this suggests that some parameters play a larger role in temporal misalignment than others.