Do Text-to-Text Multi-Task Learners Suffer from Task Conflict?

David Mueller[1,2]   **Nicholas Andrews[2]**   **Mark Dredze[1]**

[1]Department of Computer Science, Johns Hopkins University

[2]Human Language Technology Center of Excellence, Johns Hopkins University

{dam,noa}@jhu.edu   mdredze@cs.jhu.edu

Abstract

Traditional multi-task learning architectures learn a single model across multiple tasks through a shared encoder followed by task-specific decoders. Learning these models often requires specialized training algorithms that address task-conflict in the shared parameter updates, which otherwise can lead to *negative transfer*. A new type of multi-task learning within NLP homogenizes multi-task architectures as a shared encoder and language model decoder, which does surprisingly well across a range of diverse tasks (Raffel et al., 2020). Does this new architecture suffer from task-conflicts that require specialized training algorithms? We study how certain factors in the shift towards text-to-text models affects multi-task conflict and negative transfer, finding that both directional conflict and transfer are surprisingly constant across architectures.

1 Introduction

Multi-task learning (MTL) aims to obtain superior results by training a single model over multiple related tasks (Caruana, 1997). Despite MTL's promises, training neural networks with a multi-task objective often yields *worse* results on a given task than task-specific models, resulting in *negative transfer*. Why do some MTL settings work while others suffer from negative transfer? Many attribute this behavior to optimization challenges in the MTL loss surface (Ruder, 2017), that arise from *multi-task conflict*: significant differences across the gradients of different tasks which may trap SGD in poor optima (Yu et al., 2020). Prior research has focused on methods to mitigate conflict between tasks during training to ensure consistent MTL improvements (Chen et al., 2018; Sener and Koltun, 2018; Yu et al., 2020; Wang et al., 2020, *inter alia*).

Research on conflict in MTL has largely focused on the **canonical** MTL setting: all tasks share an *encoder* which projects inputs into a shared representation space, and then task-predictions are made using a *task-specific head*. Recently, it has become common in NLP to leverage a unified **text-to-text** MTL framework: all tasks are framed as sequence generation problems using a single, fully-shared language model to make predictions for each task (Figure 1; McCann et al., 2018; Raffel et al., 2020). This framing relaxes several constraints imposed in the canonical setting: text-to-text models impose almost no constraints on the output space, and leave the model to infer the task via natural language *prompts*. Despite this flexibility, text-to-text models, in combination with unsupervised pre-training, are strong multi-task learners (Raffel et al., 2020; Sanh et al., 2021).

In this work, we are interested in the following question: how does reframing MTL with a single text-to-text model affect negative (or positive) transfer, specifically with regard to multi-task conflict in the training objective? We explore this through the following contributions:

- We identify three main factors, or model properties, that change between canonical & text-to-text MTL models which may individually impact multi-task conflict and transfer.

- We empirically investigate the effect of each of these factors on negative transfer and multi-task conflict over two standard MTL benchmarks. We find that architectural factors affect multi-task conflict but have little effect on transfer.

- Finally, we show that while task prompts are necessary in multi-task learning to specify tasks, improving the semantics of task descriptions with natural language can hurt negative transfer, even if it helps zero-shot capabilities.

Overall, our findings suggest that, despite having a fully-shared parameterization between tasks, text-to-text models are surprisingly **not** inherently superior multi-task learners than canonical multi-task architectures and may similarly benefit from research in conflict mitigation techniques.
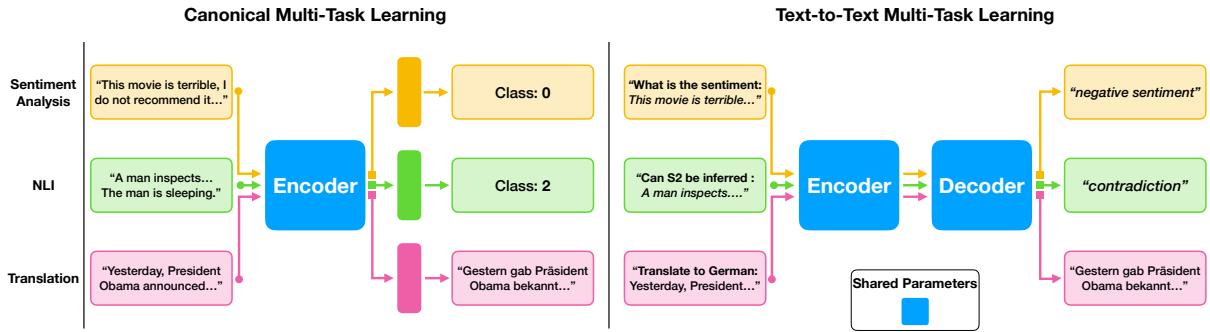
Figure 1: *Canonical* (left) vs. *Text-to-Text* (right) Multi-Task (T5) Architectures. In this work we are interested in how *negative transfer* and *multi-task conflict* are affected by moving from Canonical MTL to Text-to-Text MTL.

2 Background & Related Work

2.1 Canonical Multi-Task Learning

Suppose we have K tasks, each with a dataset $\mathcal{D}_k$ over $\mathcal{X} \times \mathcal{Y}_k$, where $\mathcal{X}$ is all sequences of tokens from a fixed vocabulary and $\mathcal{Y}_k$ is a task-specific output space. Our goal is to learn task-specific models $f_k : \mathcal{X} \to \mathcal{Y}_k$ for each task while sharing information across all tasks, without any prior knowledge about how tasks are related. Canonically, this is done by training a *shared encoder* $g_\theta : \mathcal{X} \to \mathbb{R}^d$ along with a task-specific head $h_{\phi_k} : \mathbb{R}^d \to \mathcal{Y}_k$, to project the input into a shared representation space before predicting into $\mathcal{Y}_k$.[1] Parameters θ are shared across all tasks, parameters ϕ_k are specific to task k, and we *specify* $f_k(x) \equiv (h_{\phi_k} \circ g_\theta)(x)$.

To train a canonical multi-task model, we first define task-specific loss functions $\ell_k(\hat{y}, y)$; for instance, ℓ_k may be mean-squared error for a regression task or cross-entropy for a classification task. Let $\mathcal{L}_k(\Theta)$ represent the average task loss $\ell_k(f(x; \Theta), y)$ over (x, y) pairs from a mini-batch of $\mathcal{D}_k$, where $\Theta = \{\theta, \phi_1, \ldots \phi_k\}$ for convenience. At timestep t, we update our parameters with SGD using the gradient update $\nabla_{\phi_k}\mathcal{L}_k(\Theta^t)$ for all parameters ϕ_k, and $\nabla_\theta^{MT} = \sum_{k=1}^{K} \nabla_\theta \mathcal{L}_k(\Theta^t)$ for parameters θ.

Canonical architectures have long been the dominant approach to deep multi-task learning in deep neural networks (Caruana, 1997; Ruder, 2017). Classically, multi-task conflict occurs in the multi-task gradient update ∇_θ^{MT}, and negative transfer arises from difficulties in learning a proper shared encoder model g_θ.

2.2 Text-to-Text Multi-Task Learning

In the text-to-text (T2T) MTL setting, we presuppose a function $r_k : \mathcal{Y}_k \to \mathcal{X}$ which maps the task-specific output space into natural language.[2] Given such an r_k for all tasks, our goal is to learn functions $f_k : \mathcal{X} \to \mathcal{X}$ for each task k. Because the output space for each task is the same, we can make predictions with a *shared decoder* $j_{\theta_d} : \mathbb{R}^d \to \mathcal{X}$, which yields a fully shared encoder-decoder model $f = j_{\theta_d} \circ g_{\theta_e} : \mathcal{X} \to \mathcal{X}$, where $\theta = \{\theta_e, \theta_d\}$ are shared across all tasks and g_{θ_e} is defined as above.

Given f, how do we specify f_k? A common approach is to pre-pend a fixed natural language description of the task $p_k \in \mathcal{X}$ to the input during training and testing, e.g. $f_k(x) \equiv f(p_k \frown x)$, where $\frown$ denotes concatenation. The hope of this approach is that p_k moves the input into a sufficiently distinct distribution that f learns to reliably predict task k for any input that starts with p_k.[3] This approach has shown surprising success recently: McCann et al. (2018) frame p_k as a task-specific question, and successfully train a joint Q&A model to answer every task's questions; Khashabi et al. (2020) train a single text-to-text system on 19 distinct Q&A tasks, finding that a unified input-output format is competitive with task-specific formats and systems; most recently, T5 (Raffel et al., 2020) and T0 (Sanh et al., 2021) treat **MTL as a language modeling problem**, pre-pending p_k to task inputs and jointly learning all tasks by fine-tuning a pre-trained encoder-decoder language model.

---

[1] In this work we are focused on the *hard-sharing* regime of multi-task learning. Specifically, we do not consider the *soft-sharing* regime (Ruder, 2017), which shares few to no parameters between tasks and instead transfers information across tasks in other ways.

[2] For example, if task k is abstractive summarization, where $\mathcal{Y}_k = \mathcal{X}$, then r_k is simply the identity function. If task k is a classification task, such as NLI, r_k represents a mapping of each label into a natural language description.

[3] If the training data from each task comes from sufficiently distinct distributions, the model may implicitly learn to perform each task based on the distribution of the input sentence. However, this may still be undesirable as we cannot reliably perform tasks on out-of-distribution inputs.

2.3 Negative Transfer & Multi-Task Conflict

The differentiating factor between multi-task and single-task learning is how the shared parameters are trained: ∇_θ^{MT} pulls θ towards a region of the optimization surface that might not be explored by a single-task objective. Hypothetically, this allows information to transfer across tasks yielding a stronger model (Caruana, 1997); in practice, multi-task objectives often yield *negative transfer*, underperforming single-task systems. Recent work has attributed negative transfer to *conflict* between the directions and magnitudes of task gradients. When differences between $\nabla_\theta \mathcal{L}_k(\cdot)$ across different tasks are too large, joint optimization may become difficult. As a result, prior work has focused on mitigating conflict by re-weighting task losses (Kendall et al., 2018; Chen et al., 2018; Sener and Koltun, 2018), by aligning gradient directions (Yu et al., 2020; Wang et al., 2020; Chen et al., 2020), or both (Javaloy and Valera, 2022).

Despite progress on mitigating task conflict in multi-task optimization, the relationship between conflict and negative transfer is still poorly understood. Quantifying when differences between task gradients are beneficial, and when they are conflicting, is an open problem; prior work on predicting negative transfer between tasks relies on other heuristics (Standley et al., 2020; Fifty et al., 2021; Liu et al., 2022). To that end, we study how framing multi-task learning in a text-to-text paradigm affects the relationship between tasks from the perspective of *both* task conflict and negative transfer.

2.4 Substantial Differences Between Canonical and Text-to-Text MTL

Despite rising interest in text-to-text multi-task learners (Raffel et al., 2020; Sanh et al., 2021), little attention has been given to the effects of text-to-text architectures, and especially how tasks are specified, on multi-task conflict and negative transfer. We aim to quantify the degree to which different aspects of text-to-text architectures may mitigate or exacerbate multi-task conflict, and it's impact on transfer. We identify three key factors that distinguish text-to-text MTL from canonical multi-task architectures:

F1 The head for each task is an auto-regressive language model, often with fewer constraints than task-specific heads. All tasks use the same language-modeling loss function $\ell_k(f_k(x), y) = \ell_{LM}(f_k(x), r_k(y))$.

F2 There are no task-specific parameters ϕ_k. Instead, all parameters θ are shared across all tasks, and the only gradient necessary for learning is ∇_θ^{MT}.

F3 Finally, the MTL objective depends on p_k and $r_k(\cdot)$, fixed natural language sequences which may or may not encode semantic properties of task k and it's output space.

Factors **1** & **2** reflect two key architectural decisions about the MTL model and objective, which may naturally affect multi-task conflict. For example, removing task specific parameters (Factor **2**) necessitates specifying tasks at the input; intermediate representations can no longer be task agnostic, as they now must propagate task information to the decoder. Alternatively, Factor **3** is concerned with what the model learns during training; if p_k or r_k are semantically rich, a model may *learn* to leverage them to infer task relations during training.[4] The goal of this work is to empirically study how each of these factors impacts task conflict and negative transfer in MTL.

3 Factors 1 & 2

Models We use T5 (Raffel et al., 2020), pretrained on the C4 language modeling task, as the base of our models. A pre-trained T5 model can provide rich representations of inputs which can be passed to canonical MTL heads (Ni et al., 2022) or a language model decoder for text-to-text learning. Our canonical models simply attach task-specific canonical heads to the outputs of a shared T5 encoder, and jointly fine-tune all task-specific and shared parameters. For more details on the canonical head and loss for each task, see Appendix A.

To move from canonical models to models exhibiting Factor **1**, we replace each task-specific head with a pre-trained T5 decoder head; that is, for all tasks k we replace $h_{\phi_k} : \mathbb{R}^d \rightarrow \mathcal{Y}_k$ with $j_{\phi_k} : \mathbb{R}^d \rightarrow \mathcal{X}$. Note that each decoder is still parameterized with task-specific parameters ϕ_k, and therefore each task is still specified by which decoder is used at test-time. We refer to this architecture as *Text-to-Text-ID*: models which treat all tasks as text-to-text problems, but which still leverage independent heads for task specification.[5]

---

[4]Recent work has noted that text-to-text learners rarely understand the prompts they are given during fine-tuning. See §5 for more discussion.

[5]We explore the effects of re-initializing the decoder parameters of text-to-text-ID models in Appendix C.

Architecture	Training	CoLA	SST	MSRPC	STSB	QQP	MNLI	MNLI-mm	QNLI	RTE	**Avg**
					GLUE						
Canonical	Single-Task	36.20	90.29	80.80	87.46	88.69	80.34	80.72	87.96	61.25	78.06
Canonical	Multi-Task	30.27	89.98	82.60	87.58	87.89	78.21	78.44	85.84	64.98	76.20
Text-to-Text	Single-Task	36.28	90.44	82.92	64.27	88.83	80.60	81.27	86.87	59.33	77.55
Text-to-Text-ID	Multi-Task	28.64	90.11	84.62	62.08	86.91	77.44	78.14	84.72	65.70	73.15
Text-to-Text	Multi-Task	35.12	90.71	85.21	67.92	87.51	78.98	79.73	86.30	68.47	75.55

Architecture	Training	SST	MNLI	IWSLT	CNN/DM	Seq2SQL	SQUAD	QA-SRL	QA-ZRE	Wino	**Avg**
					DecaNLP						
Canonical	Single-Task	88.74	80.40	23.55	40.22	60.28	66.57	67.34	54.85	33.93	53.41
Canonical	Multi-Task	90.92	81.15	21.77	39.48	57.25	67.29	70.07	65.79	36.90	58.96
Text-to-Text	Single-Task	89.69	80.06	23.74	40.16	60.26	73.00	63.45	47.43	27.38	56.13
Text-to-Text-ID	Multi-Task	91.61	81.10	23.05	39.88	58.52	72.53	62.90	55.57	31.85	57.44
Text-to-Text	Multi-Task	90.33	80.08	22.41	39.54	58.13	72.72	73.85	59.33	47.62	60.45

Table 1: Negative transfer between text-to-text and canonical models. Results are averaged over 3 random seeds. Multi-task model performance is highlighted in red if the performance is lower than single-task performance (negative transfer) and green if the performance is higher (positive transfer). **We can see that negative (and positive) transfer across tasks are very similar across text-to-text and canonical MTL, both at a benchmark level and an individual task level.**

Factor **2** requires a unified output space for each task in order to remove task-specific parameters. To test the effects of Factor **2** on MTL, we start from Factor **1** models (text-to-text-ID) and combine all task heads into a single decoder head shared across all tasks, j_{θ_d}. This yields a fully unified *Text-to-Text* architecture, equivalent to the original T5 MTL setting (Raffel et al., 2020). Factor **2** removes task-specific parameters from the model, and therefore we must specify tasks at the input. For the following experiments we utilize the default prompts and output spaces listed in Appendix B; in §5 we explore how Factor **3** (different task prompts and output spaces) affect our results.

Datasets We study negative transfer and multi-task conflict on two common NLP multi-task benchmarks. The first is GLUE (Wang et al., 2018), a set of 8 natural language understanding tasks. The tasks in GLUE can be separated into 7 classification tasks and one regression task; thus, canonical task heads consist of a single, linear layer that projects representations into 1, 2, or 3 dimensions. The second dataset is the NLP Decathlon (DecaNLP; McCann et al., 2018) benchmark, which frames 9 NLP tasks as Q&A problems. DecaNLP consists of span-labeling, classification, and text-to-text tasks; canonical heads for DecaNLP include linear classification layers, span-labeling classifiers, and T5 decoders for text-to-text tasks. DecaNLP therefore represents a setting with more diverse canonical heads than GLUE, making it an interesting case study against GLUE. Raffel et al. (2020) and Mc-

Cann et al. (2018) provide default p_k and r_k for GLUE and DecaNLP respectively, which we use in this section; for more details about the individual tasks of each benchmark and their default prompts see Appendix B.[6]

3.1 Negative Transfer is Similar Across Factors 1 & 2

We compare single- and multi-task models, over both canonical and text-to-text settings, in Table 1. Text-to-Text-ID models have independent heads, representing Factor **1**, and Text-to-Text multi-task models represent Factors **1** & **2**. For both GLUE and DecaNLP models, canonical and text-to-text models perform comparably on most tasks. For GLUE, text-to-text models perform marginally below canonical models, on average, notably due to STS-B, a regression task which suffers from the usage of a fixed vocabulary for numeric outputs. For DecaNLP, text-to-text models slightly outperform canonical models: 7 out of the 9 tasks are text-to-text or span-labeling tasks, the latter of which may be more amenable to text-to-text learning.

Our results illustrate that both text-to-text and canonical models exhibit similar amounts of negative and positive transfer. On GLUE, MTL reduces performance by nearly 2.5% on average in *both* text-to-text and canonical settings. In DecaNLP,

---

[6]We exclude 2 tasks from our study. The WNLI test set in GLUE is notoriously problematic for models of the size we consider, and we exclude it to avoid confusion over negative transfer. We additionally exclude the WoZ dataset in DecaNLP, because the canonical output space of WoZ is not suitable for the single-encoder canonical setup that we rely on.

MTL generally exhibits *positive transfer*—a trend also noted in the original paper (McCann et al., 2018)—suggesting that these tasks benefit each other. However, again canonical and text-to-text exhibit similar *levels* of positive transfer (a 8-10% increase in performance). Moreover, the similarities continue at the individual task-level: tasks which experience positive or negative transfer in the canonical models often experience the same amount of positive or negative transfer in text-to-text models.

We additionally see that Text-to-Text-ID models, which exhibit Factor **1**, are generally worse than Text-to-Text models, which combine Factors **1** and **2**. In tasks where this difference is significant, such as STS-B or CoLA, we identify this as an issue with the pre-trained decoder (Appendix C); re-initializing the decoders boosts the strength of STS-B and CoLA, suggesting an incompatibility between language model pre-training and regression-like tasks. For other tasks, the effect of Factor **2** is less significant: tasks such as translation appear to benefit slightly from task-specific decoders, whereas other tasks, e.g. span-labeling tasks, benefit from a shared decoder. On average, Text-to-Text models outperform Text-to-Text-ID models significantly, suggesting that the benefits of Factor **2** for some tasks outweigh the costs to other tasks in terms of performance, on the settings we consider. In short, when framing multi-task learning as a text-to-text problem (Factor **1**), it is typically better to share the decoder (Factor **2**) as well.

Despite what could be considered radically different approaches to MTL, our findings suggest that Factors **1** and **2** may not have a significant effect on the optimization landscape of multi-task learning. Factors **1** and **2** combined frame MTL as a multi-domain problem, where tasks represent distributions over the same problem space ($\mathcal{X} \rightarrow \mathcal{X}$). Our results suggest that this re-framing is not sufficient, alone, to mitigate negative transfer.[7]

4 Analyzing Multi-Task Conflict

Our results in §3.1 suggest that negative transfer is not impacted by framing MTL as a text-to-text problem. We next empirically examine whether this is borne out via *multi-task conflict*. Multi-task conflict is typically thought to be a key factor in negative transfer (§2.3). Multi-task conflict can

[7]We corroborate our results from this section on GLUE using a larger model (T5-Base) in Appendix D.

be summarized as the differences between individual task gradients; while some amount of conflict is necessary for multi-task learning to benefit over single-task learning, too much conflict can create optimization difficulties. Given that we see equivalent amounts of negative and positive transfer across canonical and text-to-text models, we expect to see similar levels of conflict as well.

Measuring Conflict We measure two types of conflict in this work: magnitude and direction. Let Θ be all the parameters of a model, $\nabla_\theta \mathcal{L}_k(\cdot)$ be the gradient of θ for task k with parameters Θ, and ∇_θ^{MT} be the *multi-task* gradient of θ with total parameters Θ (as in §2). We measure *magnitude conflict*, C_{mg}, as the variance across each task's gradient magnitude:

$$C_{mg}(\Theta) = \frac{1}{K} \sum_{k=1}^{K} \left(\left|\left| \nabla_\theta \mathcal{L}_k(\cdot) \right|\right|_2^2 - \left|\left| \nabla_\theta^{MT} \right|\right|_2^2 \right)^2$$

A high value of C_{mg} indicates high conflict between the magnitudes of task gradients, which may lead to optimization being dominated by a few high-magnitude tasks. C_{mg} is highly correlated with the magnitude of the loss at each step and the average gradient magnitude. To control for text-to-text models having training losses several orders of magnitude higher than classification losses at the beginning of training, we normalize the variance of gradient norms C_{mg} by the average gradient norm squared N_{MT}^2 at each step. Thus, when comparing gradient conflict across different models, the resulting metric measures the gradient variance if each model had the same *average* gradient norm.

We measure *directional conflict*, C_{dir}, as the norm of the multi-task gradient after normalizing ∇_θ to magnitude 1:

$$C_{dir}(\Theta) = \left|\left| \frac{1}{K} \sum_{k=1}^{K} \frac{\nabla_\theta \mathcal{L}_k(\cdot)}{||\nabla_\theta \mathcal{L}_k(\cdot)||_2} \right|\right|_2^2 - \frac{1}{K}$$

This metric is equivalent, up to a constant, to the average pairwise cosine similarity between all task gradients, which is a standard measurement for directional conflict in multi-task optimization (Yu et al., 2020; Wang et al., 2020).

Comparing conflict Our goal is to compare how task conflict changes across canonical and text-to-text models, and specifically, how Factors **1** and **2** affect task conflict. To do this, we compare C_{dir}

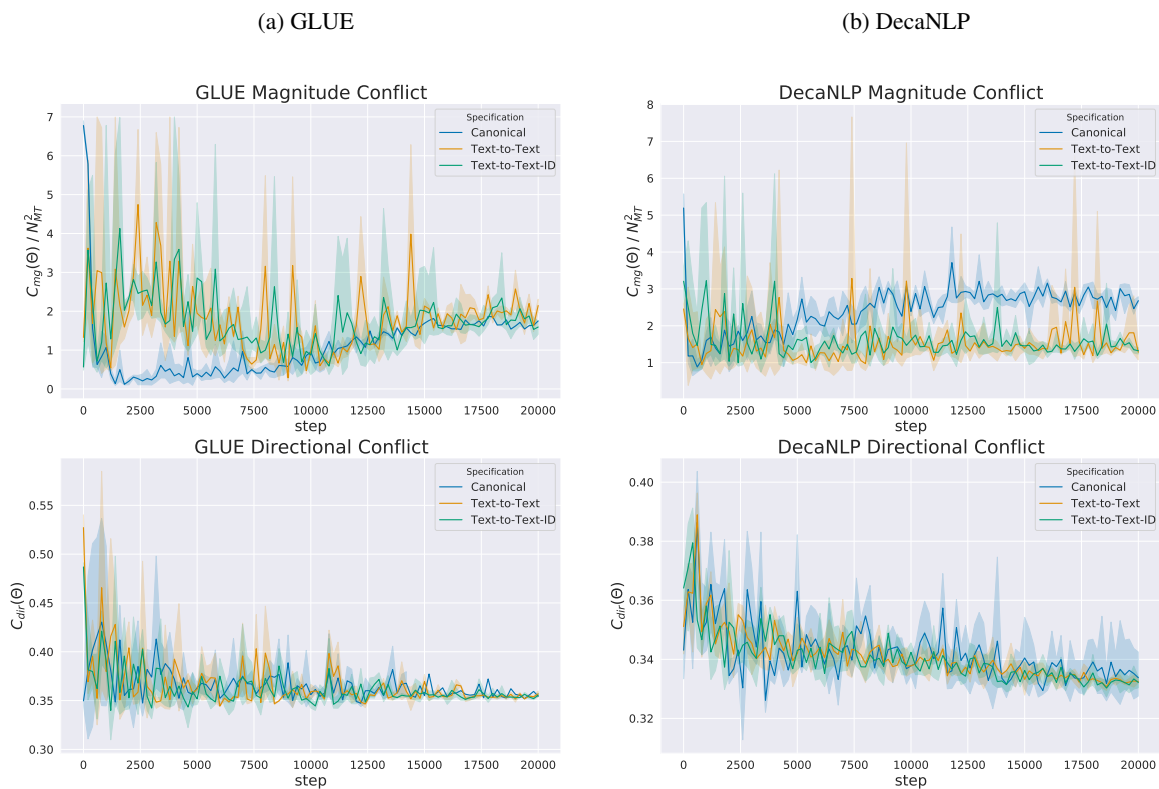|	(a) GLUE		(b) DecaNLP

Figure 2: Magnitude conflict (C_{mg}, top, normalized by average gradient magnitude) and directional conflict (C_{dir}, bottom) across GLUE (left) and DecaNLP (right) tasks in canonical, text-to-text, and text-to-text-ID models. Directional conflict in the encoder follows similar trajectories across different architectures, initially noisy and eventually stabilizing to similar values; however, magnitude conflict differs across settings, and is likely tied to the homogeneity of canonical task loss functions.

and C_{mg} in Canonical, Text-to-Text-ID, and Text-to-Text models; in our comparisons across these architectures we measure conflict in *the encoder only*, which ensures that we are measuring task-conflict across not only the same number of parameters, but over the same architecture as well (a T5 encoder).

Results We plot C_{mg} and C_{dir} across the training trajectories of canonical, text-to-text-ID, and text-to-text models in Figure 2. We see that *directional* conflict remains similar across all settings. That is, models begin training with relatively noisy average cosine similarities , and gradually over time conflict increases (pairwise cosine similarity decreases) and becomes more stable. Throughout training, text-to-text and canonical models maintain similar values of C_{dir}, and have similar variances in both GLUE and DecaNLP settings. This result corroborates our findings in §3.1: multi-task conflict, in the encoder parameters, remains consistent regardless of how tasks are specified, and this translates to similar levels of negative transfer in test-set performance.

Magnitude conflict, however, exhibits different

trends across each setting. On GLUE, canonical models have lower magnitude conflict that text-to-text models early into training, but this effect diminishes over time; eventually text-to-text and canonical models converge to similar levels of conflict. DecaNLP models exhibit a different effect: canonical models actually exhibit higher magnitude conflict than text-to-text models, which is exacerbated as training progresses. This result may be partially explained by the fact that, in GLUE, 7 out of 8 tasks share the same loss function (cross-entropy for classification), leading to more consistent magnitudes in loss towards the beginning of training; in contrast, DecaNLP canonical models compute span-labeling losses, classification losses, and conditional language model losses. Interestingly, however, the differences in *gradient* magnitude do not seem to have any bearing on negative transfer: canonical models have lower magnitude conflict in GLUE and higher magnitude conflict in DecaNLP than text-to-text models, yet both benchmarks exhibit similar transfer across canonical and text-to-text models.

Prompt	CoLA	SST	MSRPC	STSB	QQP	MNLI	MNLI-mm	QNLI	RTE	**Sum**
Single-Task (Null Prompt)	36.28	90.44	82.92	64.27	88.83	80.60	81.27	86.87	59.33	77.55
Null Prompt	29.56	90.34	83.46	48.20	86.88	76.71	76.40	85.08	50.99	50.99
Default Prompt	35.12	90.71	85.21	67.92	87.51	78.98	79.73	86.30	68.47	75.55
Diverse Prompt	32.39	89.94	79.96	66.30	87.31	77.22	77.87	86.25	65.61	73.65

Table 2: Multi-task model performance for GLUE models trained on null, default, and diverse prompts. All results are averaged over 3 random seeds. While diverse prompts may help with zero-shot transfer Sanh et al. (2021), they do not necessarily improve negative transfer in text-to-text multi-task models.

Finally, we note that conflict in the *encoder* is largely consistent between Text-to-Text and Text-to-Text-ID architectures. Therefore, we conjecture that, if multi-task conflict is beneficial when moving from independent heads to a single joint head, it is largely due to conflict that occurs in the decoder parameters.

Our findings suggest that prior work in multi-task learning which studies canonical models, especially work focused on multi-task conflict and negative transfer (§2.3), may still be relevant in text-to-text MTL. We observe that shifting from canonical to text-to-text MTL has little effect on both negative transfer and multi-task conflict. The fully-shared parameterization of text-to-text models does not translate into reductions in task conflict, nor major improvements in positive transfer between tasks. Future work on improving multi-task optimization by addressing conflict in the text-to-text setting is therefore a promising avenue.

5 Factor 3: Task Prompts & Labels

We have seen that Factors **1** & **2** have little effect on transfer and task conflict, in our settings. We now turn to Factor **3**: the form of p_k and r_k. In text-to-text MTL, tasks are specified via natural language prompts which can range in semantic richness from nondescript tokens, such as "SST: ", to descriptions of the task: "Is the sentiment of this review positive or negative?: ". In theory, a T2T multi-task model may leverage the semantics of task descriptions to implicitly align related tasks during learning, especially when leveraging a pre-trained model which exhibits non-trivial prompt understanding. However, Logan et al. (2021) demonstrated that T5 could be fine-tuned with null prompts and retain competitive accuracy, suggesting that models prefer to memorize prompts regardless of their semantic information. If T2T multi-task models memorize task prompts, the form of p_k and r_k should have no impact on task conflict.

5.1 Leveraging Diverse Prompts

Recently, Sanh et al. (2021) proposed T0, a T5 architecture fine-tuned in a massively multi-task manner with a *diverse* set of crowd-sourced prompts for each task: specifically, they leverage crowd-sourced workers to generate multiple distinct, semantically informative p_k and r_k *for each task*. Sanh et al. (2021) find that a model trained with these diverse, descriptive prompts have significantly stronger *zero-shot* capabilities than models trained on one prompt per task, suggesting T0 has learned to leverage semantic information in task prompts. This result has been called into question by Webson and Pavlick (2021), who demonstrate that few-shot performance, even with T0, can still be competitive when using nonsense prompts; however, the strength of T0's zero-shot performance suggests that diversifying the task prompts during learning may improve the model's understanding of task descriptions *to some extent*: does this correlate with less negative transfer when fine-tuning?

To test this, we leverage `promptsource`, a repository of diverse prompts for a large set of NLP tasks for text-to-text learning (Sanh et al., 2021). For each task k in GLUE, `promptsource` contains 4-7 pairs of prompts and text-labels (p_k, r_k). We re-train our T2T multi-task GLUE models using a randomly sampled pair (p_k, r_k) for each sample in task k; We term this model the *diverse* prompt model, and compare it's multi-task performance to the *default* prompt model, a model which uses task-specific tokens—similar to those in the original T5 approach—which encode no semantic information. Additionally, we train a text-to-text multi-task model with *null* prompts, e.g. no prompts. This model must learn which task it is performing by the distribution of the input samples alone.

We plot the multi-task results of null, default, and diverse T2T models in Table 2, as well as single-task text-to-text performance; we plot the multi-task conflict of models trained with the 3
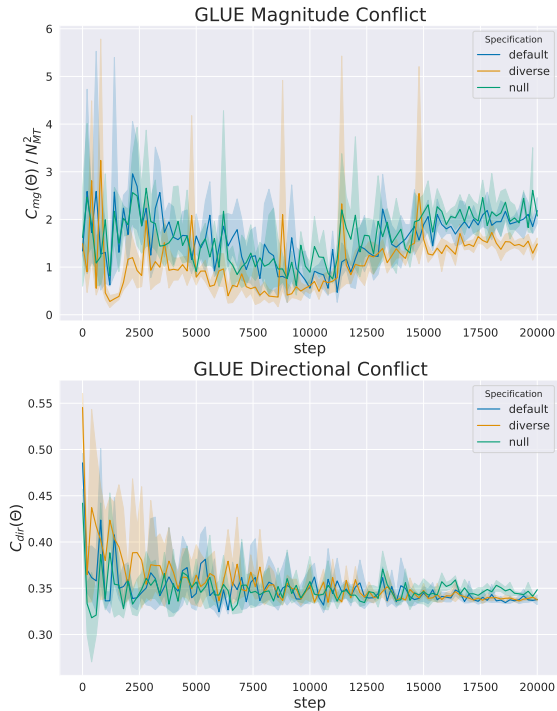
Figure 3: Magnitude conflict (C_{mg}, top) and directional conflict (C_{dir}, bottom) in GLUE models using null, default, and diverse prompts. We find that increasing prompt diversity generally *lowers* task conflict, suggesting that it pushes task distributions closer together.

different prompts in Figure 3. The cells of Table 2 are highlighted by their magnitude of positive (green) or negative (red) transfer from single-task T2T models, which are trained with null prompts. We find that null prompts perform very poorly in MTL, suggesting that T2T models struggle to determine task specification based on the input text distribution alone. Default prompts, used by the original T5 paper, which are fixed tokens pre-pended to each input, perform much better than null prompts. This result demonstrates the importance of specifying the task at the input during text-to-text MTL; while single-task models can be trained with null, and even nonsensical, prompts to competitive accuracy (Logan et al., 2021), MTL requires the text distribution to differentiate the tasks explicitly.

We additionally find that diverse prompts increase negative transfer compared to default prompts; either understanding task descriptions is not beneficial for multi-task optimization or, more likely, diversifying task prompts is not sufficient to ensure a model leverages the information in p_k during training. We note that a limitation of our work is the number of tasks we consider; it is pos-

sible that, under a MTL regime with 9 tasks, even a diverse set of prompts is not enough to improve prompt understanding. As a result, it may be that introducing multiple prompts per task serves only to confuse the model as to how tasks are specified, lowering test performance.

Finally, in Figure 3 we see that, once again, directional conflict is largely similar across training paradigms. Null prompt models see slightly less directional conflict towards the end of training, potentially due to the fact that, without task-specifications, task distributions overlap; however, lower directional conflict is *not* correlated with stronger task performance in this case. Additionally, diverse prompts appear to have consistently lower magnitude conflict throughout training; again, lower conflict is not indicative of a stronger multi-task model, as default prompt models perform the best out of the 3 specifications.

5.2 Overlap in the Output Space

§5.1 suggests that, in the settings we study, the semantics of task specifications have little effect on MTL. To that end, we test a much simpler artifact of Factor **3**: overlap across the output spaces of classification tasks. If the semantics of r_k is not important to multi-task performance, then hypothetically the selection of terms for class labels should not matter (e.g. using the letter "z" versus the term "entailment"). However, one factor that *may* impact multi-task learning is the amount of overlap in class label terms across different tasks; is it easier to learn multiple classification tasks if their label terms are distinct? To study this, we create 2 additional r_k for each classification tasks in GLUE: the first is *non-overlapping characters*, where we sample a random character for each class label, ensuring that no character is used twice across all tasks. The second is *overlapping characters*, where we instead sample a few characters, which are re-used for class labels over all tasks. Finally, we consider the default terms for class labels, which are semantically meaningful, multi-token terms.

We plot both conflict in the *decoder* of the model and average classification task accuracy, in Figure 4. We find that, similar to other factors that we study in this work, different output spaces have no observable impact on task conflict (magnitude conflict exhibits the same trend). However, we *do* observe impacts on model accuracy: models with less overlap in their output space tend to perform
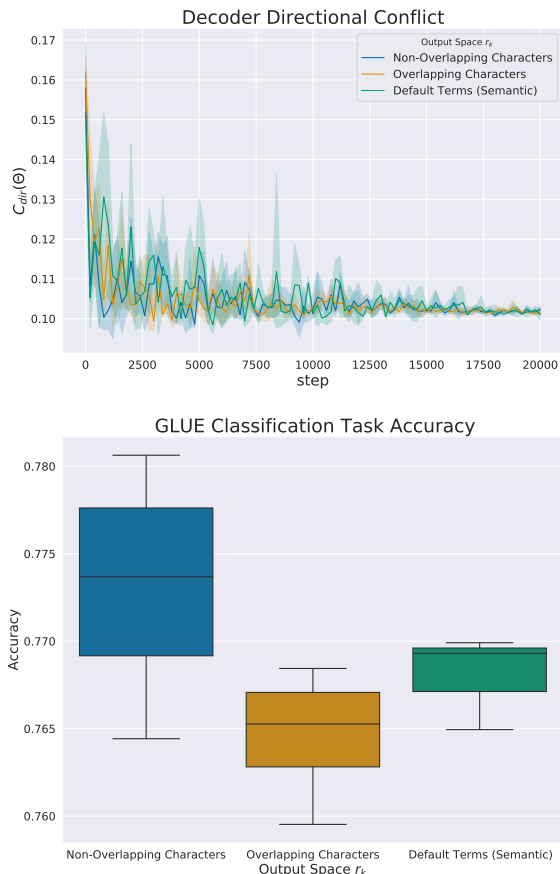
Figure 4: (Top): Directional conflict (C_{dir}) in the *decoder* of GLUE models when using different classification output spaces. The output space used, and the amount of overlap across tasks, has no observable effect on multi-task conflict. (Bottom): The impact of classification output spaces on multi-task performance. Overlap in the output space appears to have a slight effect on model accuracy: output spaces with less overlap results in higher accuracy.

better than models with more overlap, suggesting it is harmful to re-use class labels across different tasks, (although we emphasize that this may only be the case when the semantics of the class tokens are not important, i.e. in the few-task regime).

Overall, we find that it is important to specify tasks in T2T MTL: expecting the model to learn task specification from the input distribution alone leads to high negative transfer. However, when the intent is to create a strong multi-task model via fine-tuning, these prompts and outputs need not be semantically rich: simple task-specific tokens often suffice. Indeed, when MTL is being done in a setting that is *not* massively multi-task (Sanh et al., 2021; Aribandi et al., 2022), rich and diverse prompts may actually hurt multi-task learn-

ing, rather than benefit it. Additionally, throughout this work, our findings have highlighted the disconnect between standard notions of task conflict and negative transfer: task conflict, particularly across gradient magnitudes, is rarely predictive of negative transfer in the settings we study.

6 Discussion

Conclusions General purpose text-to-text architectures, typically pre-trained on large amounts of data, are increasingly popular both as a topic of academic study and in commercial NLP applications. By using natural language prompts rather than task-specific parameters, models like T5 can encode multiple tasks in the same framework with fully shared parameters. Empirically, text-to-text models often achieve comparable or superior performance compared to task-specific models. Surprisingly, we find similar task conflict in both canonical and text-to-text models, with similar negative transfer. This begs the question: are text-to-text models fundamentally better multi-task learners than canonical MTL models? Or is their success largely attributable to other factors, such as increased model capacity and pre-training on more data? Our findings suggest the latter: text-to-text architectures are **not** inherently better multi-task learners than canonical models. In particular, when controlling for other factors, we have found that T5 exhibits comparable transfer characteristics to canonical MTL parameterizations, both in terms of performance on held-out data and over the course of optimization by measuring rates of gradient conflict.

Future directions Recently, a series of optimization methods have been proposed to address the challenges posed by multi-task, or more generally, multi-objective learning (Wang et al., 2020; Javaloy and Valera, 2022). If the optimization landscape of multi-task text-to-text models poses similar challenges as canonical MTL parameterizations, applying special-purpose MTL optimization methods with text-to-text models may result in faster convergence, improvements in performance, or both. However, more research is needed both to understand the connection between task conflict and negative transfer, and to develop new optimization techniques better suited to the fully-shared parameterization used by models like T5. As such, our results suggest several promising avenues for future work, which to our knowledge have lacked clear motivation until now.

7 Limitations

A foremost limitation of our work is that we consider only a single language, English, largely due to the access of both strong text-to-text models for English as well as several English multi-task benchmarks. While we expect that our findings regarding multi-task learning are generalizable across languages, further experiments involving multi-task benchmarks and pre-trained encoders in other languages are necessary to fully support this claim.

Our work is additionally limited by the multi-task settings we examine. While we study two very popular NLP MTL benchmarks, GLUE and DecaNLP, both have limitations and prior work has shown wide variance in how well MTL works on different datasets. Notably, the canonical output space of all tasks is fairly limited, e.g. our work overall considers regression, classification, span-labeling, and sequence output spaces. However, other classical NLP output spaces are also worth studying, such as constrained structured prediction in Named Entity Recognition or Syntax Parsing. Finally, our MTL settings consider only 9 tasks to be learned jointly. A smaller universe of tasks enables us to fairly compare single- and multi-task models across several different architectures. Recent work has begun to consider *massively multi-task models* on 100+ tasks simultaneously (Aribandi et al., 2021). While we expect that our findings will generalize to these extreme settings, expanding our experiments to larger universes of tasks would validate this claim.[8]

Acknowledgements

References

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2022. Ext5: Towards extreme multi-task scaling for transfer learning. In *International Conference on Learning Representations*.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q Tran, Dara Bahri, Jianmo Ni, et al. 2021. Ext5: Towards extreme multi-task scaling for transfer learning. *arXiv preprint arXiv:2111.10952*.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Mauro Cettolo, Niehues Jan, Stüker Sebastian, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2016. The iwslt 2016 evaluation campaign.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. volume 80 of *Proceedings of Machine Learning Research*, pages 794–803, Stockholmsmässan, Stockholm Sweden. PMLR.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov. 2020. Just pick a sign: Optimizing deep multitask models with gradient sign dropout. In *Advances in Neural Information Processing Systems*, volume 33, pages 2039–2050. Curran Associates, Inc.

William B. Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. 2021. Efficiently identifying task groupings for multi-task learning. In *Advances in Neural Information Processing Systems*.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Adrián Javaloy and Isabel Valera. 2022. Rotograd: Gradient homogenization in multitask learning. In *International Conference on Learning Representations*.

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

---

[8]On the other hand, the present study is easily reproducible on commodity compute resources. Code for all of our experiments can be found at `https://github.com/davidandym/task-conflict-in-text-to-text-learners`.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, page 552–561. AAAI Press.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada. Association for Computational Linguistics.

Shikun Liu, Stephen James, Andrew Davison, and Edward Johns. 2022. Auto-lambda: Disentangling dynamic task relationships. *Transactions on Machine Learning Research*.

Robert L. Logan, Ivana Balažević, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. 2021. Cutting down on prompts and parameters: Simple few-shot learning with language models.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Cauglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 527–538. Curran Associates, Inc.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Trevor Standley, Amir Zamir, Dawn Chen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. 2020. Which tasks should be learned together in multi-task learning? In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9120–9132. PMLR.

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

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Albert Webson and Ellie Pavlick. 2021. Do prompt-based models really understand the meaning of their prompts?

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference.

Derrick Xin, Behrooz Ghorbani, Ankush Garg, Orhan Firat, and Justin Gilmer. 2022. Do current multi-task optimization methods in deep learning even help?

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Seq2SQL: Generating structured queries from natural language using reinforcement learning.

A Model and Training Details

All of our models start from the `google/t5-v1_1-small` checkpoint in the Huggingface pre-trained model hub.[9] The `t5-v1_1` model parameters are pre-trained only on C4 with the language-modeling objective, and have not been pre-trained on any multi-task data from GLUE or other datasets. It also is pre-trained with a slightly more advanced pre-training regime than the original T5 model – it uses GEGLU activation, rather than ReLU, and was pre-trained with no dropout.

Our multi-task models are fine-tuned with a learning rate of 5e-4 and a batch-size of 16, and our single-task models are fine-tuned with a learning rate of 3e-4 and a batch-size of 16. All models use a dropout rate of 0.0 and gradient clip value of 5, which we found to empirically work the best with our datasets across all settings. We fine-tune both single- and multi-task models for a total of $20,000$ steps, and use the final model for test-set predictions. We use a held-out validation set, often split from the original training data, to select all hyper-parameters mentioned above.

A.1 Canonical Architecture Decoders

Below we discuss the various task-specific heads and loss functions used for canonical models.

Regression Tasks For regression tasks we use a simple linear layer which projects sentence level representations into $\mathbb{R}$. Following (Ni et al., 2022), to compute sentence representations from T5 token encodings, we aggregate all token representations using mean pooling. For regression tasks the loss function $\mathcal{L}_k$ is mean squared error (MSE).

Classification Tasks For classification tasks we use a simple linear layer which projects sentence level representations into $\mathbb{R}^c$, where c is the number of classes. Following (Ni et al., 2022), to compute sentence representations from T5 token encodings, we aggregate all token representations using mean pooling. For regression tasks the loss function $\mathcal{L}_k$ is Cross-Entropy.

Span Labeling Tasks For span-labeling tasks we use 2 simple linear layers which project token-level representations into $\mathbb{R}$ – one layer predicts the start token, and the other predicts the end token. We aggregate token-level scores from each classifier into a single vector, and compute a softmax distribution over that vector to obtain token-level probabilities for start and end tokens. The loss function $\mathcal{L}_k$ is the average cross entropy for start and end predictions. If the task has the possibility of no answer being in the context, we additionally add a sentence-level linear binary classifier which predicts whether the context contains the answer or not, and adds an additional cross-entropy loss to the loss function.

Generative Tasks For tasks whose output space is $\mathcal{X}$, e.g. all sequences of a fixed vocabulary, we use a pre-trained $T5$ decoder. The $T5$ decoder is an autoregressive decoder, pre-trained on the C4 language modelling task. The loss function for these tasks is the same loss function as the pre-training task – $\mathcal{L}_{LM}$, the average token cross-entropy of the auto-regressive outputs.

B Data

B.1 GLUE

The GLUE dataset (Wang et al., 2018) consists of 9 tasks - 8 classification tasks and 1 regression task. In our experiments we exclude the WNLI dataset – we found that our models tended to overfit to the training data and perform below random chance on the test data.[10] This leaves 7 classification tasks and 1 regression task, which we detail below: For all datasets, we take 5% of the training dataset as a held out validation set for hyper-parameter tuning, and use the official validation sets as our held-out test sets.

CoLA is a dataset of English acceptability judgements (Warstadt et al., 2019) - each example consists of a sequence of words and is annotated with

---

wither or not the sentence is grammatical English. In GLUE, this task is treated as a *binary classification* task, with sentences classified as either acceptable or unacceptable. Our default prompt for this task, following T5, is p_k = *"cola sentence:"* and r_k = *"acceptable"* for acceptable sentences or *"unacceptable"* otherwise. The evaluation metric for this task is the Mathews correlation coefficient.

SST is a dataset of movie reviews and human annotations of their sentiment (Socher et al., 2013). The dataset is treated as a *binary classification* dataset, with sentences classified as either positive or negative. The base prompt p_k = *"sst2 sentence:"* and r_k = *"positive"* if the class is positive or *"negative"* if the class is negative. The evaluation metric for this task is accuracy.

MSRPC is a dataset of sentence pairs annotated for semantic equivalence (Dolan and Brockett, 2005) – in GLUE, this task is framed as a *binary classification*. The base prompt p_k = *"mrpc sentence 1: . . . sentence 2: . . ."* and r_k = *"not_equivalent"* if the sentences are not equivalent, and *"equivalent"* otherwise. The evaluation metric for this task is accuracy.

STS-B is a collection of sentence pairs annotated with a similarity score from 1 to 5 (Cer et al., 2017). This task is a *regression* task – a model is tasked with predicting the similarity score of the two input sentences. The base prompt p_k = *"STS sentence 1: . . . sentence 2 . . ."*. r_k simply rounds the score of each input to 1 decimal point. The evaluation metric for this task is the Spearman correlation coefficient.

QQP is a dataset of question pairs from the website Quora, annotated for whether or not the questions are semantically equivalent. The task is a *binary classification* task. The base prompt p_k = *"qqp question 1: . . . question 2: . . ."* and r_k is *"not duplicate* for non-equivalent questions, and *"duplicate"* otherwise. The evaluation metric for this task is accuracy.

MNLI & MNLI-mm is a dataset of sentence pairs with textual entailment annotations (Williams et al., 2018). The task is to determine whether the second sentence (the hypothesis) is entailed, contradicted, or neither by the first sentence (the premise). The task is a *3-class classification* task. The base prompt p_k = *"MNLI premise: . . . hypothesis: . . . ,*

and r_k is either *"entailment"*, *"neutral"*, or *"contradiction"*. The evaluation metric for this task is accuracy. The task comes with an additional mismatched test-set, MNLI-mm, which contains inputs from different domains than the training domain.

QNLI is an entailment dataset constructed from SQUAD (Rajpurkar et al., 2016) by generating question-sentence pairs from a question, and each sentence in it's corresponding context. The task is to determine whether to context sentence contains the answer to the question, making the task a *binary classification* task. Our base prompt p_k = *"qnli question: . . . sentence: . . ."*, and r_k is *"entailment"* if the context contains the answer, and *"not entailment"* otherwise. The evaluation metric for this task is accuracy.

RTE is composed of a series of textual entailment datasets constructed from news and Wikipedia text. The task is to determine whether sentence 2 is entailed by sentence 1, making the task a *binary classification* task. The base prompt p_k = *"rte sentence 1: . . . sentence 2: . . ."* and r_k is *"entailment"* is sentence 2 is entailed and *"not entailment"* otherwise. The evaluation metric for this task is accuracy.

B.2 DecaNLP

The NLP Decathlon (DecaNLP; McCann et al., 2018) frames 9 NLP tasks as question-answering problems, and proposes a novel end-to-end Q&A architecture (MQAN) to solve them. We adopt 8 of the 9 tasks for our work, which we list and briefly describe below. Unlike GLUE, DecaNLP's framing of all tasks as questions results in some tasks naturally being "self-specified" in the text-to-text framework; for example, in SQUAD, the "question" portion of the input specified to the model what task it should perform. Thus, no prompt p_k is necessary for SQUAD – the question specifies the task.

SST is the task of classifying whether or not a given sentence has a positive sentiment (Socher et al., 2013), and also appears in GLUE. This task is a *binary classification* task. In DecaNLP the base prompt p_k is *" . . . Is this review negative or positive?"*, and r_k is *"positive"* for positive reviews and *"negative"* otherwise. The evaluation metric for this task is accuracy.

MNLI is a dataset of sentence pairs with textual entailment annotations (Williams et al., 2018), and

also appears in GLUE. The task is to determine whether the second sentence (the hypothesis) is entailed, contradicted, or neither by the first sentence (the premise). The task is a *3-class classification* task. The base prompt p_k = "*Context: ... Premise: ... – entailment, neutral, or contradiction?*, and r_k is either "*entailment*", "*neutral*", or "*contradiction*". The evaluation metric for this task is accuracy.

IWSLT is a machine translation dataset - we leverage specifically the English to German IWSLT 2016 task (Cettolo et al., 2016). This task is a *sequence to sequence* task. Our base prompt is p_k = "*... Translate this sentence into german*", and r_k is the identity map. The evaluation metric for this task is BLEU.

CNN / Dailymail is a summarization dataset of news articles (Nallapati et al., 2016). The task is two generate an abstractive summary of a given input, making this a *sequence to sequence* task. Our base prompt p_k = "*... What is the summary?*", and r_k is the identity map. The evaluation metric for this task is ROUGE.

Seq2SQL is a semantic parsing dataset, where the task is to convert a natural language request into a SQL query given some additional dataset information (Zhong et al., 2018). This is a *sequence to sequence* task. Our base prompt p_k = "*... What is the translation from English to SQL?*", and r_k is the identity map. The evaluation metric for this task is Logical EM, where query outputs are converted back into database logic and evaluate at that level.

SQUAD is an extractive question answering dataset (Rajpurkar et al., 2016). For each example in squad, a context and question are given – the question is over the given context, and the answer to the question exists as a span inside the context. Thus, this task is a *span-labeling task*. The task is specified by the question, so we use no prompt for this task. r_k is simply a conversion of the span start and end points into the natural language sentence that they cover. The evaluation metric for this task is nF1.

QA-SRL is a semantic role labeling dataset – semantic role labeling is the task of assigning semantic roles, such as agent, goal, or result, to constituents of a sentence. QA-SRL (He et al., 2015) frames this task as a question and answering problem, where the question specifies a semantic role

and asks which constituent fulfills that role, and the answer is a span of the text. Thus, this task is a *span-labeling task*. The task is specified by the question, so we use no prompt for this task. r_k is simply a conversion of the span start and end points into the natural language sentence that they cover. The evaluation metric for this task is nF1.

QA-ZRE is a relation extraction dataset – the task of extracting relationships between one or more entities of a sentence. Similar to QA-SRL, QA-ZRE (Levy et al., 2017) formulates this problem as a question-answer dataset. Here the question specifies a relation, and perhaps an entity, and then asks what other entity in the sentence matches this relationship. The answer can either be a span from the sentence, or that no entity fulfills that relationship. Thus, this task is a *span-labeling task*. The task is specified by the question, so we use no prompt for this task. r_k is simply a conversion of the span start and end points into the natural language sentence that they cover, or "unanswerable" if no entity fulfills the relationship. The evaluation metric for this task is corpus-level F1.

Wino is a pronoun resolution dataset (Levesque et al., 2012), with the task being to identify which entity is being referred to by a specific pronoun in the sentence. In DecaNLP this dataset is framed as a Q&A problem task, where the question simply asks which entity in the context is being referred to by the ambiguous pronoun. In the canonical setting this can be done by labeling the entity in the text which is being referred to, yielding a *span-labeling task*. The task is specified by the question, so we use no prompt for this task. r_k is simply a conversion of the span start and end points into the natural language sentence that they cover, or "unanswerable" if no entity fulfills the relationship. The evaluation metric for this task is EM accuracy.

C   Re-Initializing Sequence Decoders

When studying the effects of Factor **1** in §3, we leverage pre-trained T5 decoders as the replacement for each task-specific head. Using T5 decoders as task-specific heads moves all task output spaces into $\mathcal{X}$, and ensures that all tasks can be trained with the same loss function ($\mathcal{L}_{LM}$). However, pre-training is *not* necessary for this shift, and could be considered a confounding factor in moving from canonical to text-to-text models, as most canonical heads do not benefit from pre-training.

		GLUE									
Architecture	Decoder	CoLA	SST	MSRPC	STSB	QQP	MNLI	MNLI-mm	QNLI	RTE	**Avg**
Text-to-Text	Pre-Trained	35.12	90.71	85.21	67.92	87.51	78.98	79.73	86.30	68.47	75.55
Text-to-Text	Re-Initialized	36.89	90.37	83.43	67.85	86.35	77.62	78.37	85.07	67.73	74.85
Text-to-Text-ID	Pre-Trained	28.64	90.11	84.62	62.08	86.91	77.44	78.14	84.72	65.70	73.15
Text-to-Text-ID	Re-Initialized	38.32	90.55	84.31	68.07	86.54	77.89	78.57	85.86	68.74	75.43

		DecaNLP									
Architecture	Decoder	SST	MNLI	IWSLT	CNN/DM	Seq2SQL	SQUAD	QA-SRL	QA-ZRE	Wino	**Avg**
Text-to-Text	Pre-Trained	90.30	79.98	22.05	39.63	57.86	72.72	74.05	59.51	48.81	60.55
Text-to-Text	Re-Initialized	88.98	75.51	9.22	27.51	42.01	66.88	56.11	53.45	22.02	49.08
Text-to-Text-ID	Pre-Trained	91.61	81.10	23.05	39.88	58.52	72.53	62.90	55.57	31.85	57.44
Text-to-Text-ID	Re-Initialized	90.04	75.32	6.40	28.37	35.22	61.87	2.76	36.92	0.79	37.52

Table 3: The effects of re-initializing the decoder head versus leveraging it's pre-trained parameters. We see that, for GLUE, re-initializing the decoder head has little effect on performance; because of the high overlap resulting from a fixed r_k for classification tasks, a strong language model is not necessary to achieve competitive results on GLUE. However, for tasks whose output spaces require generalization to *natural language* (most tasks for DecaNLP), a strong language model is necessary. Re-initializing the decoder heads for DecaNLP results in a catastrophic decrease in performance.

Here, we explore the effects of *re-initializing* the T5 decoder heads, removing the biases of pre-training when modeling the output space of each task. In Table 3 we plot multi-task model performance on both GLUE and DecaNLP using Pre-trained and Re-Initialized decoder heads, for both Text-to-Text and Text-to-Text-ID architectures. We additionally plot the average task loss during training of all models in Figure 5.

Perhaps unsurprisingly, the effects of re-initializing the decoder on GLUE are small: in both Text-to-Text and Text-to-Text-ID models, performance is similar when using re-initialized or pre-trained decoders. Two notable exceptions are CoLA and STS-B, both regression-like tasks; when trained with independent heads, performance is *significantly* higher when using a re-initialized decoder. This result may suggest that the biases of a pre-trained decoder head are ill-suited for regression-like tasks. Interestingly, when task-heads are combined across tasks this discrepancy largely disappears: Text-to-Text architectures outperform Text-to-Text-ID architectures when using pre-trained decoders. Overall, the effects of re-initialization on performance are reflected in the training trajectories: in Figure 5 (top) we see that re-initializing the decoder has almost no effect on model convergence for GLUE.

However, the results of re-initializing the decoder on DecaNLP are significantly negative. 7 of the 9 tasks in DecaNLP are span-labeling or sequence generation tasks; unlike classification tasks, these 7 tasks contain extremely diverse output spaces, requiring either extractive or abstractive sequence generation. Because good generalization on these tasks requires a strong language model, re-initializing the decoder on this benchmark is disastrous for most tasks considered; not only is performance significantly affected, but we can see from Figure 5 that re-initialized decoders fail to even converge to good minima. The only exceptions are the SST and MNLI tasks, which are both classification tasks.

D Increasing Model Capacity

Finally, we explore the effects of increasing model size on our GLUE results. All experiments in the main paper use the T5-Small architecture, which contains far fewer parameters than current SotA text-to-text models. As a small corroborative experiment, we study how expanding the model size from T5-Small to T5-Base affects our main results.

In Table 4 we replicate our core experiment (studying negative transfer across Canonical & Text-to-Text architectures) using a pre-trained T5-Base model as the base of our architectures. We find that increasing the model size does not have a significant effect on our key conclusion: negative transfer at the task-level remains similar in canonical and text-to-text settings. However, as noted by Xin et al. (2022), we find that increasing the model size alleviates negative transfer on the whole: T5-Small exhibits negative transfer at the benchmark level for GLUE (Table 1), whereas T5-Base benefits from positive transfer when considering the average task performance.

| Architecture | Decoder | GLUE | | | | | | | | | |
		CoLA	SST	MSRPC	STSB	QQP	MNLI	MNLI-mm	QNLI	RTE	Avg
Canonical	Single-Task	42.93	92.49	81.91	88.69	90.04	84.97	85.56	89.68	64.33	78.76
Canonical	Multi-Task	40.52	92.16	84.89	88.09	89.27	82.49	83.15	88.93	70.28	79.97
Text-to-Text	Single-Task	43.75	92.16	82.21	66.48	90.11	85.45	85.76	90.86	63.90	77.56
Text-to-Text	Multi-Task	45.83	92.59	86.91	67.92	89.45	84.12	84.76	89.85	73.36	79.42

Table 4: Negative transfer between text-to-text and canonical models with the T5-Base architecture (a larger model than used in the main paper), averaged over 5 random seeds. T5-Base largely corroborates our findings, exhibiting similar amounts of negative & positive transfer across canonical and text-to-text architectures.
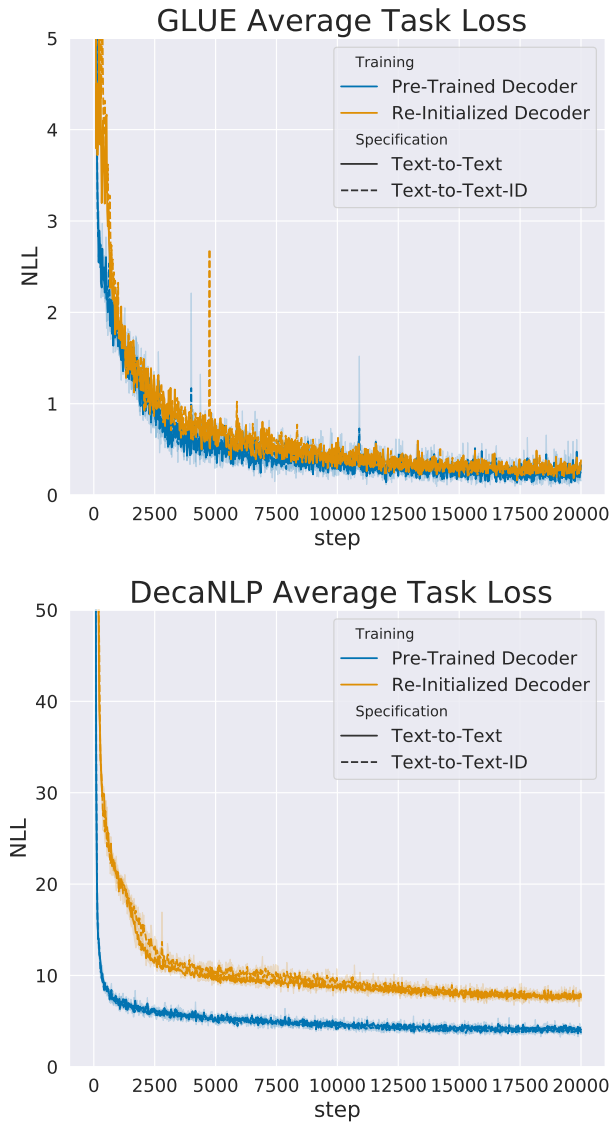

Figure 5: The average task training loss of GLUE (top) and DecaNLP (bottom) for Text-to-Text and Text-to-Text-ID architectures, using pre-trained and re-initialized decoders. For GLUE, re-initializing the decoder has little effect on the convergence of text-to-text models: all models converge to equivalently optimal minima. However, for DecaNLP, re-initialized decoders are unable to fit the data as well as pre-trained decoders.