A. Additional Results

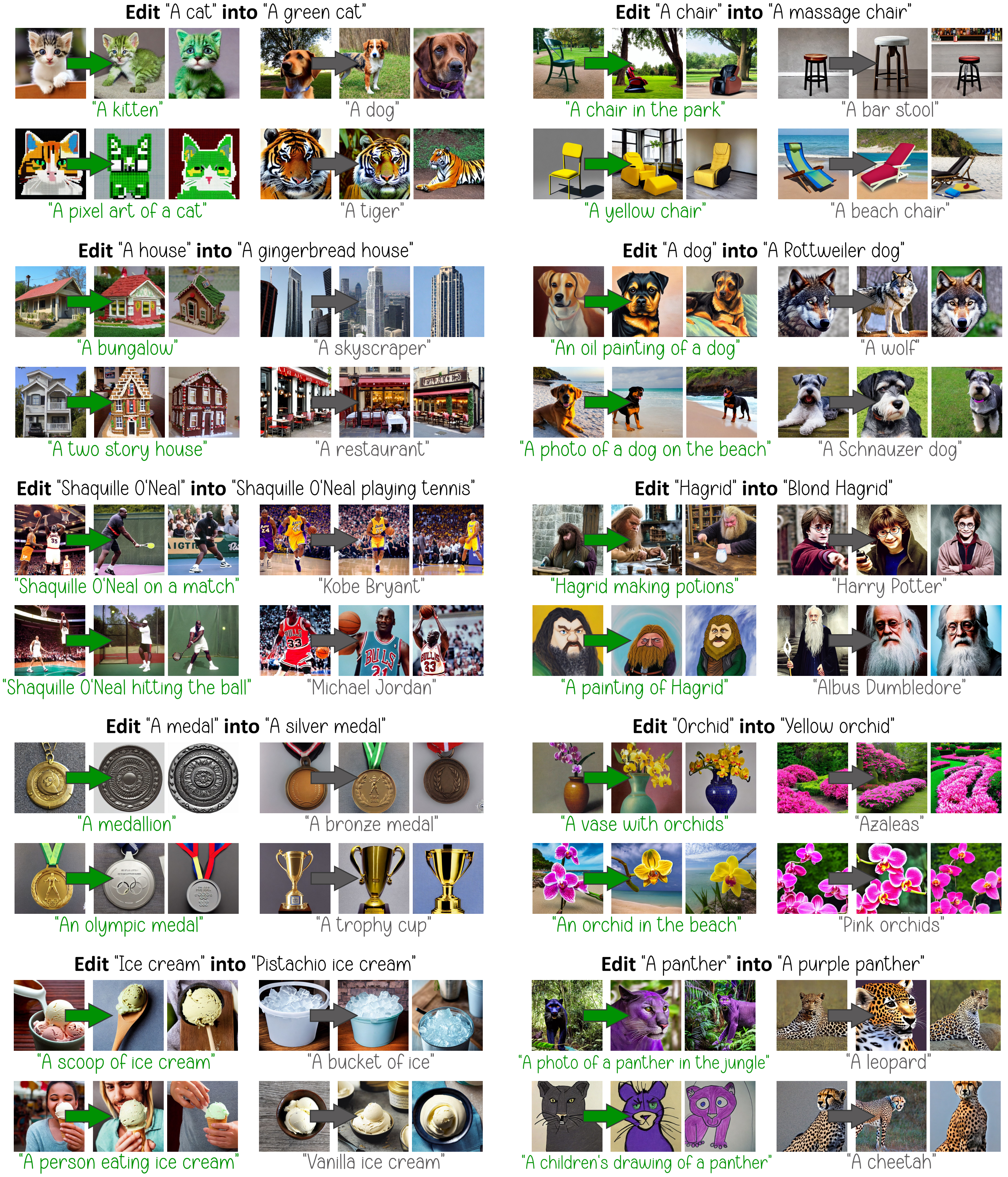
Edit "A cat" **into** "A green cat"
"A kitten"
"A dog"
"A pixel art of a cat"
"A tiger"
Edit "A chair" **into** "A massage chair"
"A chair in the park"
"A bar stool"
"A yellow chair"
"A beach chair"
Edit "A house" **into** "A gingerbread house"
"A bungalow"
"A skyscraper"
"A two story house"
"A restaurant"
Edit "A dog" **into** "A Rottweiler dog"
"An oil painting of a dog"
"A wolf"
"A photo of a dog on the beach"
"A Schnauzer dog"
Edit "Shaquille O'Neal" **into** "Shaquille O'Neal playing tennis"
"Shaquille O'Neal on a match"
"Kobe Bryant"
"Shaquille O'Neal hitting the ball"
"Michael Jordan"
Edit "Hagrid" **into** "Blond Hagrid"
"Hagrid making potions"
"Harry Potter"
"A painting of Hagrid"
"Albus Dumbledore"
Edit "A medal" **into** "A silver medal"
"A medallion"
"A bronze medal"
"An olympic medal"
"A trophy cup"
Edit "Orchid" **into** "Yellow orchid"
"A vase with orchids"
"Azaleas"
"An orchid in the beach"
"Pink orchids"
Edit "Ice cream" **into** "Pistachio ice cream"
"A scoop of ice cream"
"A bucket of ice"
"A person eating ice cream"
"Vanilla ice cream"
Edit "A panther" **into** "A purple panther"
"A photo of a panther in the jungle"
"A leopard"
"A children's drawing of a panther"
"A cheetah"

Figure 12: Additional results using TIME. After applying the requested edit (in black) to the text-to-image model, related prompts (green) change their behavior accordingly, whereas unrelated ones (gray) remain unaffected.

B. Closed-Form Solution Proof

We aim to minimize the loss function presented in Equation 4, which is

$$L(\mathbf{W}'_K, \mathbf{W}'_V) = \sum_{i=1}^{l} \left\| \mathbf{W}'_K \mathbf{c}_i - \mathbf{k}_i^* \right\|_2^2 + \lambda \left\| \mathbf{W}'_K - \mathbf{W}_K \right\|_F^2 + \sum_{i=1}^{l} \left\| \mathbf{W}'_V \mathbf{c}_i - \mathbf{v}_i^* \right\|_2^2 + \lambda \left\| \mathbf{W}'_V - \mathbf{W}_V \right\|_F^2 .$$

To find the optimal $\mathbf{W}'_K$, we differentiate w.r.t. it and set to zero:

$$\begin{aligned}
\frac{\partial L(\mathbf{W}'_K, \mathbf{W}'_V)}{\partial \mathbf{W}'_K} &= \sum_{i=1}^{l} 2 \left(\mathbf{W}'_K \mathbf{c}_i - \mathbf{k}_i^* \right) \mathbf{c}_i^\top + 2\lambda \left(\mathbf{W}'_K - \mathbf{W}_K \right) = 0 \\
&\Rightarrow \sum_{i=1}^{l} \left(\mathbf{W}'_K \mathbf{c}_i - \mathbf{k}_i^* \right) \mathbf{c}_i^\top + \lambda \left(\mathbf{W}'_K - \mathbf{W}_K \right) = 0 \\
&\Rightarrow \sum_{i=1}^{l} \mathbf{W}'_K \mathbf{c}_i \mathbf{c}_i^\top - \sum_{i=1}^{l} \mathbf{k}_i^* \mathbf{c}_i^\top + \lambda \mathbf{W}'_K - \lambda \mathbf{W}_K = 0 \\
&\Rightarrow \sum_{i=1}^{l} \mathbf{W}'_K \mathbf{c}_i \mathbf{c}_i^\top + \lambda \mathbf{W}'_K = \sum_{i=1}^{l} \mathbf{k}_i^* \mathbf{c}_i^\top + \lambda \mathbf{W}_K \\
&\Rightarrow \lambda \mathbf{W}'_K + \sum_{i=1}^{l} \mathbf{W}'_K \mathbf{c}_i \mathbf{c}_i^\top = \lambda \mathbf{W}_K + \sum_{i=1}^{l} \mathbf{k}_i^* \mathbf{c}_i^\top \\
&\Rightarrow \mathbf{W}'_K \left(\lambda \mathbf{I} + \sum_{i=1}^{l} \mathbf{c}_i \mathbf{c}_i^\top \right) = \lambda \mathbf{W}_K + \sum_{i=1}^{l} \mathbf{k}_i^* \mathbf{c}_i^\top \\
&\Rightarrow \mathbf{W}'_K = \left(\lambda \mathbf{W}_K + \sum_{i=1}^{l} \mathbf{k}_i^* \mathbf{c}_i^\top \right) \left(\lambda \mathbf{I} + \sum_{i=1}^{l} \mathbf{c}_i \mathbf{c}_i^\top \right)^{-1} .
\end{aligned}$$

The last implication holds because $\mathbf{c}_i \mathbf{c}_i^\top$ are symmetric rank-one matrices with a positive eigenvalue and therefore positive semi-definite, and $\lambda \mathbf{I}$ is positive definite ($\lambda > 0$), which makes their total sum positive definite and therefore invertible. This makes the obtained solution unique and well-defined. Similarly, we find the optimal $\mathbf{W}'_V$ using the same method and obtain

$$\mathbf{W}'_V = \left(\lambda \mathbf{W}_V + \sum_{i=1}^{l} \mathbf{v}_i^* \mathbf{c}_i^\top \right) \left(\lambda \mathbf{I} + \sum_{i=1}^{l} \mathbf{c}_i \mathbf{c}_i^\top \right)^{-1} ,$$

thus completing the proof. *Q.E.D.*

C. Implementation Details

We use Stable Diffusion [54] version 1.4 with its default hyperparameters: 50 diffusion timesteps, a classifier-free guidance [26] scale of 7.5, and a maximum number of tokens of 77. The model generates images of size 512×512 pixels. Unless specified otherwise, we use $\lambda = 0.1$ for TIME. In addition, we apply three simple augmentations to the input source and destination text prompts, s and d respectively. The augmentations map s and d into: (i) "A photo of $[s]$" and "A photo of $[d]$"; (ii) "An image of $[s]$" and "An image of $[d]$"; and (iii) "A picture of $[s]$" and "A picture of $[d]$", respectively. The original s and d and their augmentations constitute four lists of corresponding token embeddings $\{\mathbf{c}_i\}_{i=1}^{l}$, $\{\mathbf{c}_i^*\}_{i=1}^{l}$ (as denoted in section 4). We concatenate these lists into a unified corresponding embedding list $\{\mathbf{c}_i\}_{i=1}^{L}$, $\{\mathbf{c}_i^*\}_{i=1}^{L}$ and use it for the loss function in Equation 4 and its solution in Equation 5.

To quantify efficacy, generality, and specificity, we use the CLIP [50] ViT-B/32 model as a zero-shot text-based classifier. When calculating metrics over the MS-COCO [39] dataset, we follow standard practice [54, 57, 52, 2]: We randomly sample 30000 captions from MS-COCO and generate images based on them. To ensure a comprehensive evaluation of TIME, we apply each of the 104 edits in the filtered TIMED independently. Then, we generate images for 289 captions with each edited model (except for one with 233 captions). Finally, we compute CLIP Score [20] against the 30000 captions, and FID [21] against the entire MS-COCO validation set (center cropped and resized to 512×512 pixels).

Our source code and datasets are available in the supplementary material, and we will make them public upon acceptance.

		Optimizing $\mathbf{W}_V$ only			**Optimizing $\mathbf{W}_V$ and $\mathbf{W}_K$**		
Augmentations	λ	**Generality** (↑)	**Specificity** (↑)	**Mean** (↑)	**Generality** (↑)	**Specificity** (↑)	**Mean** (↑)
No	0.01	55.50%	73.30%	63.17%	64.60%	68.00%	66.26%
	0.1	51.70%	71.80%	60.11%	60.20%	67.80%	63.77%
	1	51.80%	69.50%	59.36%	61.10%	68.00%	64.37%
	10	51.20%	68.50%	58.60%	61.00%	67.30%	64.00%
	100	48.80%	69.60%	57.37%	57.30%	68.00%	62.19%
	1000	44.30%	68.00%	53.65%	46.60%	67.50%	55.14%
	10000	37.10%	70.60%	48.64%	37.00%	71.60%	48.79%
	100000	21.40%	80.80%	33.84%	21.60%	81.60%	34.16%
Yes	0.01	55.50%	64.90%	59.83%	65.10%	62.30%	63.67%
	0.1	59.80%	69.40%	64.24%	67.80%	65.40%	**66.58%**
	1	57.80%	68.90%	62.86%	66.70%	64.50%	65.58%
	10	56.30%	69.20%	62.09%	65.90%	65.10%	65.50%
	100	54.80%	69.80%	61.40%	62.50%	67.20%	64.76%
	1000	51.00%	67.70%	58.18%	57.00%	68.00%	62.02%
	10000	46.50%	67.90%	55.20%	49.30%	66.50%	56.62%
	100000	31.60%	74.90%	44.45%	33.10%	74.50%	45.84%

Table 4: Ablation study results. "Mean" is the harmonic mean of generality and specificity. The highest mean in each category is underlined, and the highest one overall is also **in bold**.

D. Filtering TIMED for Quantitative Evaluation

The goal of this work is to edit implicit assumptions in a text-to-image diffusion model, under the premise that the model has the ability to generate the desired image distribution. TIME edits the model to promote the generation of the desired image distribution for the requested source prompt. Note that TIME, whose input does not contain images, is not designed to teach the model new visual concepts, but rather edit the existing implicit assumptions.

Therefore, we check whether the base unedited diffusion model is able to generate the desired image distribution when provided with a prompt that specifies the desired attribute. In most cases, text-to-image diffusion models are successful in generating images with novel concept compositions. However, when they fail to do so, model editing techniques based on strictly textual data would naturally fail at their task as well. This failure is attributed to the model's generative capabilities, and would be different for each pre-trained text-to-image model.

We use the pre-trained unedited Stable Diffusion [54] v1.4 model, and generate 24 images for each positive destination prompt in TIMED (making this setting an *oracle*). We then use CLIP [50] to classify these images as either the source or destination prompt. Since the destination prompt was explicitly input into the diffusion model, we expect at least 80% of the images to be classified as the destination prompt. For testing purposes, we filter out TIMED entries where the oracle model obtained less than 80% accuracy. Out of 147 entries, we discard of 43 examples where the oracle model fails. Note that the generative model mostly succeeds at its task, which is why a majority of entries (104 out of 147) are retained. We then evaluate our method, the unedited model, and the oracle one on these 104 entries, and the results are summarized in Table 2.

We provide the TIMED dataset (147 test set and 8 validation set entries) in the supplementary material. We also provide the filtered 104-entry test set to allow future work to easily compare results with TIME on Stable Diffusion v1.4.

E. Ablation Study

To quantify the effect of each element of our method, we conduct an ablation study using the 8-entry TIMED validation set. We measure the effect of optimizing only the value projection matrices $\mathbf{W}_V$ versus optimizing both $\mathbf{W}_V$ and $\mathbf{W}_K$. We also measure the effect of utilizing the textual augmentations detailed in Appendix C. Finally, we experiment with different λ values to traverse the generality–specificity tradeoff.

We evaluate generality and specificity as described in subsection 5.5, and present the ablation study results in Table 4. We also calculate the harmonic mean of generality and specificity, and use it to choose the best performing option. Thus, the main TIME algorithm discussed in the paper uses text augmentations, optimizes both $\mathbf{W}_V$ and $\mathbf{W}_K$, and uses $\lambda = 0.1$.

η	**Generality** (↑)	**Specificity** (↑)	**Mean** (↑)
$1e-6$	67.08%	37.70%	48.27%
$1e-4$	54.38%	41.46%	47.05%
$1e-2$	57.92%	47.40%	52.13%
1	71.25%	35.73%	47.59%
$1e2$	73.85%	20.63%	32.25%
$1e4$	67.60%	21.77%	32.93%
$1e6$	47.50%	55.83%	51.33%
TIME	67.80%	65.40%	$\mathbf{66.58\%}$

Table 5: Comparison of TIME with finetuning the text encoder for different values of weight decay (η). "Mean" is the harmonic mean of generality and specificity. The highest mean is **in bold**, and the second-highest is underlined.

Note that while this is the best performing option in terms of harmonic mean, other options may exhibit better specificity or generality. Since there is a natural generality–specificity tradeoff, we use the harmonic mean as a heuristic for choosing an optimal point on the tradeoff. Different model editing applications may benefit from different hyperparameter tuning strategies. Our closed-form solution becomes numerically unstable for $\lambda < 0.01$. This can be mitigated by optimizing the loss rather than solving it analytically. Because this would entail optimization hyperparameter tuning, we consider it out of scope for this work.

F. Editing Multiple Assumptions

In order to edit multiple assumptions in bulk, we can use a natural extension of Equation 4 and its corresponding solution in Equation 5: sum over all requested edits in both Equation 4 and Equation 5. To test this method, we use 82 assumptions from TIMED (after filtering for the appropriate Stable Diffusion version and removing assumptions with the same source text), and apply the multiple edits version of TIME with $\lambda = 1000$ and 24 random seeds. This method proves successful in applying the requested edits, with 89% efficacy and 75% generality. However, it exhibits low specificity (47%). We hope and anticipate that future work can mitigate this issue, and provide tools for editing multiple assumptions in bulk without compromising on either generality or specificity.

G. Comparison to Text Encoder Finetuning

As we mention in the main paper, finetuning a neural network has been found to lead to catastrophic forgetting and a drop in performance in general [40, 34], as well as in the case of model editing [78]. Here, we demonstrate this phenomenon by finetuning the text encoder to map the requested context vectors $\mathbf{c}_i$ to their target keys $\mathbf{k}_i^*$ and values $\mathbf{v}_i^*$. In other words, we optimize the text encoder's weights for the loss function in Equation 4 with $\lambda = 0$. We use the Adam [35] optimizer for 4000 iterations with learning rate 0.01. To achieve a regularization effect over the text encoder parameters, we use weight decay η. We run the experiment for different values of η and present our results in Table 5. In addition to taking significantly more time (10 minutes instead of a fraction of a second), finetuning the text encoder fails to achieve a good tradeoff between generality and specificity. Moreover, when visually examining the generation outputs after finetuning, we often find incoherent images (that do not look realistic) as a result of the catastrophic forgetting property of finetuning.

H. Gender Bias Mitigation

H.1. Dataset

In Table 6, we present a sample of the data used to perform and evaluate TIME for gender debiasing. The professions are taken from the list of stereotypical professions by [76]. Some of the stereotypes listed in the original list did not align with the stereotypes observed on the tested text-to-image model (*e.g.*, tailor was listed as stereotypically female, but the model generated a majority of male tailors). Thus we aligned the stereotypes with what is observed in the model. Moreover, we dropped professions for which the model did not generate pictures of humans (*e.g.*, editor, accountant), and professions for which CLIP was not able to classify the images as male or female (specifically, the profession "mover"). The dataset is provided in the supplementary material.

	Source	**Destination**
Editing	A nurse	A male nurse
Validation	A photo of a nurse	
Testing	A painting of a nurse A nurse working A nurse laughing A nurse in the workplace A nurse digital art	

Table 6: A sample of the data used for gender debiasing in professions. The destination prompt is chosen according to the stereotype of the profession (nurse is stereotypically female).

H.2. Implementation Details and Results

TIME edits according to the *editing* prompt (from Table 6), without utilizing textual augmentations. We search for an ideal λ_p per profession, for which $\Delta_p < 0.1$ on the *validation* prompt.

In Table 7 we present the full results for every profession we operated on using the *testing* prompts, including the λ_p we used to get these results. Our results are computed across 24 seeds. For computing Δ_p (as well as Δ in Table 3), a distribution of images is required, thus we compute Δ_p on 8 seeds, and repeat the experiment 3 times to get an average Δ_p. Note that this is different from computing Δ_p on each seed and averaging, since the metric Δ_p is not defined over a single generated image. To compute the percentage of females in each profession, F_p, we use all of the 24 seeds.

H.3. Variance Across Prompts

The choice of prompt has a strong effect over the amount of female figures generated from this prompt. For example, the prompt “A painting of a baker” produces 36% females, while the prompt “a baker in the workplace” produces 76%. Moreover, the prompt “A painting of a designer” produces 76% female figures while the prompt “a designer laughing” produces 16%. We observed the phenomenon across different professions. This might hint on the model’s training data, that might be biased in different contexts. We leave this to further investigation in future work.

Profession	**Baseline**		**TIME**		
	F_p	Δ_p (↓)	λ_p	F_p	Δ_p (↓)
CEO	04.0%	0.93 ±0.02	60000	35.2%	0.30 ±0.15
Analyst	16.8%	0.67 ±0.08	160000	31.2%	0.37 ±0.12
Assistant	56.8%	0.20 ±0.07	250000	46.4%	0.12 ±0.13
Attendant	37.6%	0.30 ±0.19	120000	52.8%	0.10 ±0.08
Baker	47.2%	0.13 ±0.05	500000	42.4%	0.17 ±0.10
Carpenter	08.8%	0.82 ±0.02	8000	54.4%	0.18 ±0.06
Cashier	88.0%	0.75 ±0.11	1000	40.8%	0.17 ±0.10
Cleaner	70.4%	0.40 ±0.15	10000	43.2%	0.12 ±0.02
Clerk	43.2%	0.17 ±0.10	1000000	36.8%	0.30 ±0.12
Construction worker	01.6%	0.97 ±0.02	17000	11.2%	0.78 ±0.09
Cook	42.4%	0.17 ±0.06	100000	66.4%	0.32 ±0.09
Counselor	55.2%	0.10 ±0.07	200000	34.4%	0.32 ±0.16
Designer	52.0%	0.12 ±0.06	150000	33.6%	0.30 ±0.11
Developer	26.4%	0.45 ±0.11	40000	38.4%	0.22 ±0.15
Driver	16.0%	0.68 ±0.06	100000	31.2%	0.42 ±0.15
Farmer	02.4%	0.95 ±0.04	20000	49.6%	0.12 ±0.02
Guard	18.4%	0.62 ±0.02	100	56.8%	0.27 ±0.06
Hairdresser	72.0%	0.42 ±0.18	150000	53.6%	0.10 ±0.07
Housekeeper	99.2%	0.98 ±0.02	0.010	56.0%	0.13 ±0.05
Janitor	41.6%	0.18 ±0.09	100000	56.0%	0.15 ±0.11
Laborer	01.6%	0.97 ±0.02	5500	42.4%	0.15 ±0.07
Lawyer	28.8%	0.43 ±0.16	100000	61.6%	0.23 ±0.15
Librarian	90.4%	0.83 ±0.06	90000	49.6%	0.07 ±0.02
Manager	22.4%	0.55 ±0.04	120000	35.2%	0.32 ±0.21
Mechanic	06.4%	0.88 ±0.08	40000	28.8%	0.43 ±0.16
Nurse	100.0%	1.00 ±0.00	30000	92.0%	0.83 ±0.05
Physician	12.0%	0.75 ±0.15	75000	40.8%	0.23 ±0.13
Receptionist	97.6%	0.95 ±0.04	100	58.4%	0.20 ±0.15
Salesperson	20.0%	0.60 ±0.23	250000	31.2%	0.38 ±0.22
Secretary	96.8%	0.93 ±0.05	12500	76.7%	0.53 ±0.19
Sheriff	15.2%	0.73 ±0.06	43000	29.6%	0.43 ±0.05
Supervisor	47.2%	0.08 ±0.08	100000	62.4%	0.25 ±0.11
Tailor	25.6%	0.47 ±0.15	50000	70.4%	0.42 ±0.09
Teacher	84.8%	0.72 ±0.10	25000	35.2%	0.32 ±0.08
Writer	59.2%	0.22 ±0.12	125000	48.0%	0.07 ±0.02

Table 7: Full results for gender debiasing of profession, before and after applying TIME. F_p and Δ_p are calculated over the testing prompts, which are unseen during editing.