Toward Semantic History Compression for Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

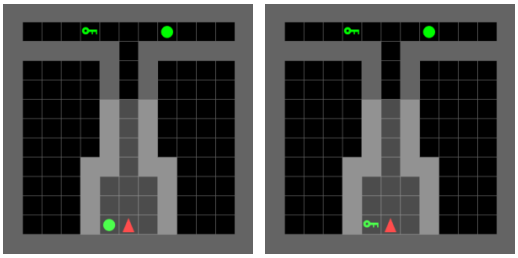
Abstract

Agents interacting under partial observability require access to past observations via a memory mechanism in order to approximate the true state of the environment. Recent work suggests that leveraging language as abstraction provides benefits for creating a representation of past events. History Compression via Language Models (HELM) leverages a pretrained Language Model (LM) for representing the past. It relies on a randomized attention mechanism to translate environment observations to token embeddings. In this work, we show that the representations resulting from this attention mechanism can collapse under certain conditions. This causes blindness of the agent to certain subtleties in the environment. We propose a solution to this problem consisting of two parts. First, we improve upon HELM by substituting the attention mechanism with a feature-wise centering-and-scaling operation. Second, we take a step toward semantic history compression by encoding the observations with a pretrained multimodal model such as CLIP, which further improves performance. With these improvements our model is able to solve the challenging MiniGrid-Memory environment. Surprisingly, however, our experiments suggest that this is not due to the semantic enrichment of the representation presented to the LM but only due to the discriminative power provided by CLIP.

1 Introduction

In Reinforcement Learning (RL) an agent interacts with an environment and learns from feedback provided in the form of a reward function. RL agents that are deployed in the real world often have to cope with partial observability. Therefore, the capability to approximate the true state of their surrounding environment by virtue of an agent's perception is crucial (Åström, 1964; Kaelbling et al., 1998). To this end, many agents employ a memory mechanism to track events that occurred in the past. In this memory, it is much more efficient to store abstract representations of the past rather than every detail the agent encountered. Thus, memory mechanisms such as LSTM (Hochreiter & Schmidhuber, 1997) or Transformer (Vaswani et al., 2017) compress sequences of high-dimensional observations.

Similarly, also humans memorize abstract concepts rather than every detail of information they encountered in the past. In humans, the ability to abstract is heavily influenced by the exposure to language in early childhood (Waxman & Markow, 1995). By design, language is very well suited to form *abstractions* and it is used on a daily basis to pass on information between humans. Therefore, it is a natural choice as a medium for compressing compounding information. Prior work has illustrated that pretrained LMs can efficiently compress sequences of observations and facilitate agent learning in partially observable RL environments (HELM, Paischer et al., 2022). The key challenge hereby is to map image-based observations to language representations. HELM tackles this problem using a randomly chosen mapping called FrozenHopfield (FH) that is — due to its



β	Hellinger distance δ
1	$0.00036 \pm 1.88\text{e-}9$
10	$0.0036 \pm 1.84\text{e-}7$
100	$0.035 \pm 2.09\text{e-}5$
1000	0.091 ± 0.01
1e4	0.093 ± 0.06

Figure 1: **Left:** Instances of memory environment. The agent only observes the white shaded region, and must navigate to the object seen in the starting room. **Right:** Hellinger distance (Hellinger, 1909) of softmax distributions over token embeddings for the two observations of the Memory environment on the left. Mean over 1000 different initializations of $\boldsymbol{P}$ of FH is shown. For very high values of $\beta > 1\text{e}4$, the softmax converges to a one-hot encoding and δ directly corresponds to the probability of mapping to a single, but different token embedding. Therefore, with higher values of β, there is only a 10% chance of avoiding collapse to the same token embedding.

randomness — inherently unable to form meaningful abstractions. In fact, we show that under certain conditions, the representations are prone to collapse rendering the agent unable to distinguish between different inputs (see Figure Fig. 1). We build upon HELM by (i) substituting FH with a feature-wise centering-and-scaling operation and (ii) incorporating a CLIP image encoder (Radford et al., 2021), which is pretrained in a multimodal fashion on web data consisting of images and text. We term the resulting new method HELMv2. Using HELMv2, we are able to distinguish even minute differences in the input when necessary, which drastically enhances downstream performance.

We demonstrate the effectiveness of HELMv2 on a diverse set of partially observable environments. Concretely, we train on 2D MiniGrid (Chevalier-Boisvert et al., 2018), and 3D MiniWorld environments (Chevalier-Boisvert, 2018). HELMv2 yields significant improvements over HELM in all environments. Further, we conduct ablation studies, which show that the improvements are not only due to the CLIP image encoder but also due to the replacement of the FH. Finally, we construct a mapping that successfully conveys the semantics extracted by CLIP to the LM. Surprisingly, however, using this mapping does not further improve the results in the selected environments.

2 Methods

HELM has demonstrated that pretrained LMs are well suited for compressing past observations that are randomly mapped to language tokens, which is realized by FH. The FH mechanism consists of a random matrix $\boldsymbol{P}$ and an attention mechanism over pretrained token embeddings (VocabAttn). More formally, let $\boldsymbol{E} = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_k)^\top \in \mathbb{R}^{k \times m}$ be the token embedding matrix of the pretrained LM consisting of k embeddings $\boldsymbol{e}_i \in \mathbb{R}^m$. At every timestep t, we obtain inputs $\boldsymbol{x}_t \in \mathbb{R}^m$ for the LM from observations $\boldsymbol{o}_t \in \mathbb{R}^n$ via the FH mechanism by

$$\boldsymbol{x}_t^\top = \sigma(\beta \boldsymbol{o}_t^\top \boldsymbol{P}^\top \boldsymbol{E}^\top)\boldsymbol{E}, \tag{1}$$

where σ is the softmax function and $\boldsymbol{P} \in \mathbb{R}^{m \times n}$ has entries sampled from $\mathcal{N}(0, n/m)$. The resulting $\boldsymbol{x}_t$ lies in the convex hull of the token embeddings of the LM. The parameter β is a scaling factor that controls the dispersion of $\boldsymbol{x}_t$ within that convex hull.

HELMcs Our aim is to avoid representation collapse caused by the FH mechanism. We illustrate in Fig. 1 that FH is prone to representation collapse even for higher values of β, especially if observations tend to be visually similar. To sidestep this issue, we substitute the attention mechanism in FH with a feature-wise centering-and-scaling operation. Let $\mathcal{B}$ be a batch of observations $\boldsymbol{o}_t$ and let $\boldsymbol{\mu}_\mathcal{B} \in \mathbb{R}^n$ be its mean feature vector and $\boldsymbol{\sigma}_\mathcal{B} \in \mathbb{R}^n$ the vector of standard deviations both taken over $\mathcal{B}$. Likewise let $\boldsymbol{\mu}_E \in \mathbb{R}^m$ and $\boldsymbol{\sigma}_E \in \mathbb{R}^m$ be means and standard deviations of the token embeddings $\boldsymbol{E}$. Then we compute $\boldsymbol{x}_t$ as

$$\boldsymbol{x}_t = \operatorname{diag}(\boldsymbol{\sigma}_E)\boldsymbol{P}\operatorname{diag}(\boldsymbol{\sigma}_\mathcal{B})^{-1}(\boldsymbol{o}_t - \boldsymbol{\mu}_\mathcal{B}) + \boldsymbol{\mu}_E, \tag{2}$$

where diag takes a vector and constructs a diagonal matrix from it. We refer to this setting as HELMcs.

HELMv2 We obtain another setting by encoding the observations with the ResNet-50 CLIP image encoder, which has the same output dimension as the embeddings $\boldsymbol{e}_i$ used by TransformerXL (TrXL, Dai et al., 2019), which we use as LM, thus eliminating the need for the mapping $\boldsymbol{P}$. That is, we let $\boldsymbol{z}_t = \text{CLIP}(\boldsymbol{o}_t) \in \mathbb{R}^m$ and construct $\mathcal{B}$ from $\boldsymbol{z}_t$. Consequently, $\boldsymbol{\mu}_{\mathcal{B}} \in \mathbb{R}^m$ and $\boldsymbol{\sigma}_{\mathcal{B}} \in \mathbb{R}^m$. We compute the LM inputs as

$$\boldsymbol{x}_t = \text{diag}(\boldsymbol{\sigma}_E)\,\text{diag}(\boldsymbol{\sigma}_{\mathcal{B}})^{-1}(\boldsymbol{z}_t - \boldsymbol{\mu}_{\mathcal{B}}) + \boldsymbol{\mu}_E. \tag{3}$$

3 Experimental Results

We investigate the limitations of HELM on a memory environment (Memory, Fig. 1, left). The task for the agent is to remember the object in the room it spawns in and navigate through a corridor ending in a T-junction and containing two objects, one of which is equivalent to the object in the starting room. If the agent chooses the wrong direction at the junction the episode ends without eliciting any reward. However, the objects only slightly differ in shape (green ball vs green key). We demonstrate that FH collapses to the same representation for both objects by measuring the distances between the softmax distribution $\sigma(\cdot)$ over token embeddings (Fig. 1, right). Indeed, we observe that even for high values of β there is very little chance that HELM can discriminate between the two objects.

We measure the cosine similarity and ℓ_2-distances between the embeddings of both observations from HELM, HELMcs, and HELMv2. Compared to HELM and HELMcs, HELMv2 is able to separate the two observations much better (see Table 1).

Method	Cosine Similarity	ℓ_2-distances
HELM, $\beta = 1$	$0.99 \pm$ 1.68-7	0.005 ± 0.002
HELM, $\beta = 10$	$0.99 \pm$ 5.65e-6	0.048 ± 0.022
HELM, $\beta = 100$	$0.99 \pm$ 8.2e-4	0.57 ± 0.38
HELM, $\beta = 1000$	0.97 ± 0.08	5.2 ± 9.75
HELMcs	$0.99 \pm$ 5.98e-5	3.48 ± 0.59
HELMv2	0.906	26.25

Table 1: Average cosine similarities and ℓ_2-distances for HELM, HELMcs, and HELMv2. HELMv2 can better discriminate between the two different objects.

We compare HELMv2 to HELM on a set of partially observable environments (see Fig. 2). Particularly, we select the RedBlueDoors environment (MiniGrid-RedBlueDoors-6x6-v0) from Chevalier-Boisvert et al. (2018) , and the TMaze environment (MiniWorld-TMaze-v0) from Chevalier-Boisvert (2018). Furthermore, we show results for the Memory environment (MiniGrid-MemoryS11-v0) as shown in Fig. 1. HELMv2 significantly outperforms HELM on all environments ($p =$ 3.84e-4, $p =$ 2.25e-11, $p =$ 9.19e-9, for Memory, RedBlueDoors, TMaze, respectively). Due to its limitations, HELM does not achieve performance better than randomly choosing an object for the Memory environment. Contrary, HELMv2 is able to distinguish and memorize the objects and solve this task. On the 3D TMaze environment HELMv2 shows drastically improved performance compared to HELM. Additionally, we conduct an ablation study to show how VocabAttn affects the performance of HELM. Indeed, we illustrate that the attention over token embeddings consistently results in a drop in performance in Appendix C Fig. 4.

All our methods are trained with Proximal Policy Optimization (PPO, Schulman et al., 2017). We evaluate all methods via the interquantile mean (IQM) and 95% bootstrapped confidence intervals (CIs) as suggested by Agarwal et al. (2021). To test for statistical significance, we perform a Wilcoxon test (Wilcoxon, 1945) at the end of training. We follow the architectural design of Paischer et al. (2022) and perform the same hyperparameter search for all our environments.

4 Future Work and Discussion

In this work, we want to pave the way toward a semantic history compression for RL, that is to preserve the semantic concepts of an observation when mapping it into the LM space. In this regard,

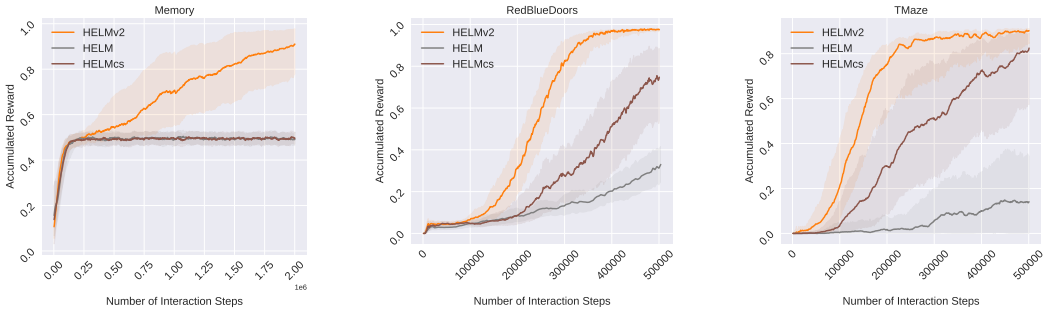


Figure 2: IQM and 95% bootstrapped CIs across 15 seeds for Memory (**left**), and 30 seeds for RedBlueDoors (**middle**), and TMaze (**right**). HELMv2 consistently outperforms HELM

we provide compelling evidence that it is indeed feasible to create such a semantic mapping between CLIP and the LM (see Fig. 3). When using this semantic mapping in the two selected minimalistic RL environments, however, we do not observe a significant improvement over HELMv2.

We can compute such a mapping from the vocabularies of the CLIP language encoder $\mathcal{V}_{\text{CLIP}}$ and the LM encoder $\mathcal{V}_{\text{LM}}$ alone. First, we identify the overlap between the two vocabularies $\mathcal{V}_{OV} = \mathcal{V}_{\text{CLIP}} \cap \mathcal{V}_{\text{LM}}$ with size $l = |\mathcal{V}_{OV}|$. Next, we embed $\mathcal{V}_{OV}$ in the CLIP output space and the LM input space yielding embedding matrices $\boldsymbol{F} = (\boldsymbol{f}_1, \dots, \boldsymbol{f}_l) \in \mathbb{R}^{n \times l}$ and $\boldsymbol{E}_l \in \mathbb{R}^{m \times l}$, respectively. We compute an orthogonal linear mapping matrix $\boldsymbol{W}$ according to the Procrustes method, which is commonly used for aligning monolingual embedding spaces (Artetxe et al., 2018; Hoshen & Wolf, 2018; Smith et al., 2017; Lample et al., 2018; Zhang et al., 2016; Xing et al., 2015; Minixhofer et al., 2022). We refer to this setting as SHELM (for semantic HELM) and elaborate in more detail on it in Appendix A.

Due to the alignment of the CLIP image encoder and the CLIP language encoder, this mapping can be used to project images to the LM space while preserving its semantics. We visualize sample images and the closest tokens in the LM space in Fig. 3. Further, in Appendix B we conduct a quantitative analysis using publicly available image captioning datasets and show that our mapping can identify tokens prevalent in captions provided by humans. Table 5 in Appendix C corroborates these impressions.

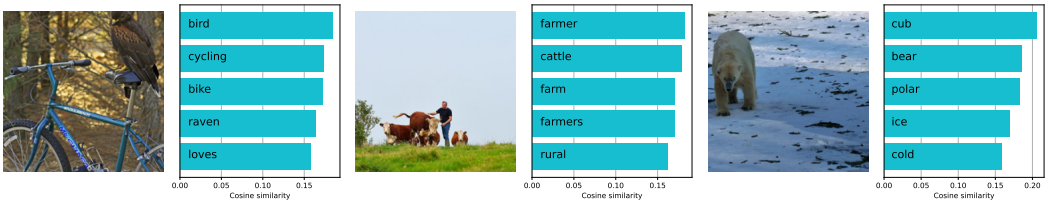


Figure 3: Top 5 closest tokens in the LM embedding space after applying the linear mapping to an image embedded by CLIP.

We show preliminary results of SHELM in Fig. 5. Our findings are that SHELM improves upon HELM, since it avoids the apparent representation collapse in the FH mechanism due to the orthogonal mapping $\boldsymbol{W}$. However, we observe performance on-par with HELMv2 (0.979 ± 0.002 for RedBlueDoors, 0.887 ± 0.018 for TMaze). We believe this is due to the fact that our selected environments are artificial 2D and 3D scenes for which the CLIP image encoder is unable to sufficiently extract semantically meaningful features. This might change in real world environments with observations close to the CLIP pre-training domain. We aim to investigate this question in future work.

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. G. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W. (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 29304–29320, 2021.

Artetxe, M., Labaka, G., and Agirre, E. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2289–2294, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1250.

Artetxe, M., Labaka, G., and Agirre, E. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 789–798, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1073.

Chevalier-Boisvert, M. MiniWorld: Minimalistic 3D Environment for RL & Robotics Research, 2018. Publication Title: GitHub repository.

Chevalier-Boisvert, M., Willems, L., and Pal, S. Minimalistic Gridworld Environment for OpenAI Gym, 2018. Publication Title: GitHub repository.

Craswell, N. Mean Reciprocal Rank. In LIU, L. and ÖZSU, M. T. (eds.), *Encyclopedia of Database Systems*, pp. 1703–1703. Springer US, Boston, MA, 2009. doi: 10.1007/978-0-387-39940-9_488.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V., and Salakhutdinov, R. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In Korhonen, A., Traum, D. R., and Màrquez, L. (eds.), *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pp. 2978–2988. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1285.

Gower, J. and Dijksterhuis, G. Procrustes Problems. *Procrustes Problems, Oxford Statistical Science Series*, Vol. 30, January 2005. doi: 10.1093/acprof:oso/9780198510581.001.0001. ISBN: 9780198510581.

Groenen, P. J. F., Giaquinto, P., and Kiers, H. A. L. An improved majorization algorithm for robust procrustes analysis. *Springer US*, pp. 151–158, 2005.

Hellinger, E. Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen. *Journal für die reine und angewandte Mathematik*, 1909(136):210–271, 1909. doi: doi:10.1515/crll.1909.136.210.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

Hoshen, Y. and Wolf, L. Non-Adversarial Unsupervised Word Translation. In Riloff, E., Chiang, D., Hockenmaier, J., and Tsujii, J. (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 469–478. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-1043.

Järvelin, K. and Kekäläinen, J. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Trans. Inf. Syst.*, 20(4):422–446, October 2002. ISSN 1046-8188. doi: 10.1145/582415.582418. Place: New York, NY, USA Publisher: Association for Computing Machinery.

Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. Planning and Acting in Partially Observable Stochastic Domains. *Artificial Intelligence*, 101:99–134, 1998.

Lample, G., Conneau, A., Ranzato, M., Denoyer, L., and Jégou, H. Word translation without parallel data. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Lin, T.-Y., Maire, M., Belongie, S. J., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft COCO: Common Objects in Context. In Fleet, D. J., Pajdla, T., Schiele, B., and Tuytelaars, T. (eds.), *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pp. 740–755. Springer, 2014. doi: 10.1007/978-3-319-10602-1_48.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Burges, C. J., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Minixhofer, B., Paischer, F., and Rekabsaz, N. WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models. In Carpuat, M., Marneffe, M.-C. d., and Ruíz, I. V. M. (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pp. 3992–4006. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.naacl-main.293.

Paischer, F., Adler, T., Patil, V. P., Bitto-Nemling, A., Holzleitner, M., Lehner, S., Eghbal-zadeh, H., and Hochreiter, S. History Compression via Language Models in Reinforcement Learning. *CoRR*, abs/2205.12258, 2022. doi: 10.48550/arXiv.2205.12258. arXiv: 2205.12258.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning Transferable Visual Models From Natural Language Supervision. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal Policy Optimization Algorithms. *CoRR*, abs/1707.06347, 2017. arXiv: 1707.06347.

Schönemann, P. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31(1): 1–10, 1966.

Smith, S. L., Turban, D. H. P., Hamblin, S., and Hammerla, N. Y. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

Stewart, G. W. The Efficient Generation of Random Orthogonal Matrices with an Application to Condition Estimators. *SIAM Journal on Numerical Analysis*, 17(3):403–409, 1980. doi: 10.1137/0717034. _eprint: https://doi.org/10.1137/0717034.

Tewel, Y., Shalev, Y., Schwartz, I., and Wolf, L. Zero-Shot Image-to-Text Generation for Visual-Semantic Arithmetic. *arXiv preprint arXiv:2111.14447*, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

Waxman, S. R. and Markow, D. Words as Invitations to Form Categories: Evidence from 12- to 13-Month-Old Infants. *Cognitive Psychology*, 29:257–302, 1995.

Wilcoxon, F. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, 1945. ISSN 00994987. Publisher: [International Biometric Society, Wiley].

Xing, C., Wang, D., Liu, C., and Lin, Y. Normalized Word Embedding and Orthogonal Transform for Bilingual Word Translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1006–1011, Denver, Colorado, May 2015. Association for Computational Linguistics. doi: 10.3115/v1/N15-1104.

Zhang, Y., Gaddy, D., Barzilay, R., and Jaakkola, T. Ten Pairs to Tag – Multilingual POS Tagging via Coarse Mapping between Embeddings. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1307–1317, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1156.

Åström, K. J. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10:174–205, 1964.

A Linear Mappings between Language Embedding Spaces

Our aim is to create a linear mapping $\boldsymbol{W}$ from the aligned multimodal space of CLIP to the embedding space of a pretrained LM encoder. We argue that a linear mapping is sufficient since the semantic relation between the tokens is equivalent in both spaces and semantic relations between tokens can be expressed additively (Mikolov et al., 2013; Tewel et al., 2021). Therefore, given two CLIP tokens $\boldsymbol{f}_1, \boldsymbol{f}_2$, it does not matter whether we perform the addition before or after the mapping, i.e., $\boldsymbol{W}(\boldsymbol{f}_1 + \boldsymbol{f}_2) = \boldsymbol{W}\boldsymbol{f}_1 + \boldsymbol{W}\boldsymbol{f}_2$. Further, using linear models to map between monolingual word embedding spaces is established in the literature (Artetxe et al., 2018; Hoshen & Wolf, 2018; Smith et al., 2017; Lample et al., 2018; Zhang et al., 2016; Xing et al., 2015; Minixhofer et al., 2022).

Algorithm 1 Associating CLIP output space with LM input space

Require: CLIP language encoder CLIP_{LM}, Vocabulary of CLIP $\mathcal{V}_{\text{CLIP}}$, Language Model Encoder $\text{LM}(\boldsymbol{v}_i)$, Language Model vocabulary $\mathcal{V}_{\text{LM}}$

$\mathcal{V}_{\text{OV}} \leftarrow \mathcal{V}_{\text{CLIP}} \cap \mathcal{V}_{\text{LM}}$ ▷ Search for overlapping vocabulary

$\boldsymbol{f}_i = \text{CLIP}_{LM}(\boldsymbol{v}_i) \quad \text{for} \quad \boldsymbol{v}_i \in \mathcal{V}_{\text{OV}}$ ▷ Embed tokens in CLIP output space

$\boldsymbol{e}_i = \text{LM}(\boldsymbol{v}_i) \quad \text{for} \quad \boldsymbol{v}_i \in \mathcal{V}_{\text{OV}}$ ▷ Embed tokens in LM input space

$\boldsymbol{W} \leftarrow \texttt{create_mapping}(\boldsymbol{F}, \boldsymbol{E})$ ▷ Compute mapping between embeddings

We define the vocabulary of the CLIP language encoder as $\mathcal{V}_{\text{CLIP}}$ and the vocabulary of the LM encoder as $\mathcal{V}_{\text{LM}}$. First, we search for an overlap between the two vocabularies and retrieve the indices of the tokens in the overlap $\mathcal{V}_{OV}$, where $l = |\mathcal{V}_{OV}|$. Next, we embed the tokens in the CLIP output space to obtain a token embedding matrix $\boldsymbol{F} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_l) \in \mathbb{R}^{n \times l}$. Let $\boldsymbol{E} = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_l) \in \mathbb{R}^{m \times l}$ be the token embedding matrix from the pretrained LM consisting of l embeddings $\boldsymbol{e}_i \in \mathbb{R}^m$ corresponding to tokens in $\mathcal{V}_{OV}$. We compute a linear mapping matrix $\boldsymbol{W}$ that minimizes the distance between the two token embedding matrices $\|\boldsymbol{W}\boldsymbol{F} - \boldsymbol{E}\|_F^2$. In principle any linear model can be applied to map from the CLIP space to the LM space. Algorithm 1 shows the algorithm for computing the mapping used for SHELM.

The prevalent alignment of modalities in the CLIP output space allows mapping visual inputs to the LM space. In turn we can substitute the FH mechanism in HELM with CLIP followed by our mapping $\boldsymbol{W}$ to preserve the semantics of an observation in the LM space.

We compare four different types of linear models:

- Ordinary Least Squares (Linear), minimizing
$$\arg\min_{\boldsymbol{W}} \|\boldsymbol{F}\boldsymbol{W} - \boldsymbol{E}\|_F^2 \tag{4}$$
- Least Squares with Thikonov regularization (Ridge):
$$\arg\min_{\boldsymbol{W}} \|\boldsymbol{F}\boldsymbol{W} - \boldsymbol{E}\|_F^2 + \lambda\|\boldsymbol{W}\|_F^2 \tag{5}$$
- Orthogonal Procrustes (Least Squares with orthogonality constraint, Schönemann (1966); Gower & Dijksterhuis (2005)):
$$\arg\min_{\boldsymbol{W}} \|\boldsymbol{F}\boldsymbol{W} - \boldsymbol{E}\|_F^2 \quad \text{subject to} \quad \boldsymbol{W}^\top\boldsymbol{W} = \boldsymbol{I} \tag{6}$$
- Robust Procrustes (RobProc, (Groenen et al., 2005)) iteratively refines the Orthogonal Procrustes method based on the predicion error $\|\boldsymbol{F}\boldsymbol{W} - \boldsymbol{E}\|_F^2$[1].

[1]For the detailed algorithm, we refer the reader to Groenen et al. (2005)

	Linear	Ridge	Procrustes	RobProc
RN50	0.786/0.0013	0.571/0.281	0.64/**0.304**	0.65/0.303
RN101	0.561/0.013	0.56/0.277	0.621/0.313	0.642/**0.314**
RN50x4	0.628/0.003	0.625/0.27	0.678/**0.329**	0.691/0.326
RN50x16	0.694/0.002	0.692/0.272	0.722/0.345	0.732/**0.348**
RN50x64	0.798/0.001	0.79/0.274	0.742/**0.356**	0.745/0.353
ViT-B/32	0.576/0.008	0.574/0.295	0.595/**0.321**	0.612/0.317
ViT-B/16	0.585/0.004	0.583/0.309	0.614/**0.342**	0.631/0.337
ViT-L/14	0.705/0.002	0.677/0.314	0.669/**0.368**	0.677/0.367
ViT-L/14*	0.705/0.003	0.677/0.321	0.666/**0.37**	0.676/0.367

Table 2: Train/Test accuracy for different linear models optimized for mapping CLIP tokens to the TrXL embedding space. Average over 5-fold cross validation is shown. Centering and scaling is used for pre-processing of the token embeddings. For the Procrustes mappings, zero-padding is applied if the CLIP embedding is of lower dimensionality than the TrXL embeddings. The Procrustes methods yield the best mapping between the two embedding spaces. ViT-L/14* received images resized to 336 pixels as input during pretraining. Linear heavily overfits to the training set. Larger backbones lead to a better test accuracy.

The Procrustes method is commonly used for aligning embedding spaces across different languages since it preserves *monolingual invariance* (Artetxe et al., 2016; Minixhofer et al., 2022). We consider various choices of CLIP backbones and align their output spaces with the TrXL used in HELM. We determine the intersection of the vocabulary of the CLIP language encoder and the TrXL language encoder. In total, there are 5285 tokens that appear in both, $\mathcal{V}_{\text{CLIP}}$, and $\mathcal{V}_{\text{LM}}$. Prior work has found that as little as ten word correspondences are sufficient to train an orthogonal mapping between monolingual embedding spaces of closely related languages (Zhang et al., 2016). This assumes a certain degree of isomorphism between the embedding spaces. Since we train a mapping between embedding spaces of the same language, we expect this assumption to hold in our setting as well. The CLIP output space and the TrXL input space differ greatly in their statistics, therefore we perform centering and scaling as preprocessing. This way, during inference in the RL experiments we can simply re-center and re-scale by the statistics of LM space to obtain a suitable input for the TrXL. We perform a 5-fold cross validation and measure the accuracy, considering each token as its own class. Table 2 shows the average train and test accuracy for the different linear mapping methods for various CLIP backbones and TrXL. We observe a strong overfitting effect for Linear, while the two Procrustes variants generalize best.

B Mapping Visual Input to the Language Space

We conduct a quantitative analysis of the different mapping methods. In this regard, we use a pre-existing image/caption dataset to quantify how well the semantics of an image are preserved after mapping to the LM space. We draw a random subset of 1000 image-caption pairs of the popular MSCOCO dataset (Lin et al., 2014). The subset is filtered to contain only image-text pairs where the captions contain at least 5 tokens of our computed vocabulary overlap in Appendix A. For preprocessing of the captions we remove stop words and apply stemming. Next, we propagate the image through various CLIP backbones, to obtain an image embedding and map it to the LM space using our pre-computed mappings. Finally, we rank tokens in the LM space based on their cosine similarity to the mapped image. Based on the obtained ranking we compute the Mean Reciprocal Rank (MRR, Craswell, 2009) and the Normalized Discounted Cumulative Gain (NDCG, Järvelin & Kekäläinen, 2002). We also include a comparison to token ranking in the CLIP space as a baseline to determine how much information is lost after applying our mapping.

Table 3 shows the NDCG for different linear mappings trained to map between RN50 CLIP backbone and TrXL. The results for MRR are shown in Table 4. We observe that the Robust Procrustes mapping consistently outperforms the Linear and Ridge mapping. Further, there is a slight improvement for larger model architectures. Remarkably, on average the third ranked token is contained in a caption

for the Procrustes mapping, while for CLIP it is every second ranked token. From this analysis we conclude that most of the semantics are preserved with our linear mapping.

	Linear	Ridge	Procrustes	RobProc	CLIP
RN50	0.252±0.038	0.285±0.062	0.325±0.082	**0.326±0.082**	0.421±0.099
RN101	0.26±0.042	0.285±0.06	0.344±0.089	**0.345±0.09**	0.523±0.099
RN50x4	0.255±0.037	0.275±0.049	0.345±0.089	**0.347±0.089**	0.42±0.096
RN50x16	0.253±0.037	0.272±0.052	0.34±0.089	**0.343±0.089**	0.419±0.099
RN50x64	0.257±0.042	0.265±0.047	0.342±0.091	**0.343±0.091**	0.429±0.101
ViT-B/32	0.255±0.038	0.269±0.052	0.35±0.088	**0.35±0.089**	0.433±0.102
ViT-B/16	0.251±0.039	0.278±0.056	0.357±0.091	**0.356±0.091**	0.435±0.0.099
ViT-L/14	0.252±0.037	0.271±0.049	0.35±0.093	**0.351±0.093**	0.43±0.098
ViT-L/14*	0.255±0.041	0.271±0.05	0.356±0.095	**0.357±0.097**	0.439±0.099

Table 3: Normalized Discounted Cumulative Gain (NDCG) for ranked tokens in the LM embedding space given an image as input and applying our mapping. Image-caption pairs are drawn from the MSCOCO dataset. Ranking tokens in the CLIP space provides an upper bound. ViT-L/14* receives images resized to 336 pixels as input.

	Linear	Ridge	Procrustes	RobProc	CLIP
RN50	0.034±0.113	0.115±0.231	0.26±0.35	**0.264±0.353**	0.514±0.4
RN101	0.054±0.144	0.102±0.21	0.315±0.366	**0.318±0.37**	0.532±0.393
RN50x4	0.03±0.086	0.077±0.179	0.322±0.37	**0.33±0.374**	0.524±0.391
RN50x16	0.023±0.059	0.078±0.185	0.308±0.372	**0.322±0.379**	0.52±0.4
RN50x64	0.05±0.153	0.062±0.17	0.311±0.376	**0.314±0.375**	0.532±0.0.396
ViT-B/32	0.032±0.095	0.073±0.189	0.329±0.369	**0.332±0.372**	0.558±0.405
ViT-B/16	0.036±0.127	0.102±0.23	0.371±0.384	**0.372±0.39**	0.554±0.4
ViT-L/14	0.036±0.107	0.073±0.182	0.329±0.381	**0.337±0.382**	0.552±0.4
ViT-L/14*	0.043±0.129	0.069±0.164	0.352±0.386	**0.362±0.392**	0.575±0.393

Table 4: Mean Reciprocal Rank (MRR) for ranked tokens in the LM embedding space given an image as input and applying our mapping. On average, the first token appearing in an image caption is approximately ranked third. Image-caption pairs are drawn from the MSCOCO dataset. MRR in the CLIP output space serves as upper bound. On average CLIP ranks the first token appearing in a corresponding caption on second position. ViT-L/14* receives images resized to 336 pixels as input.

C Additional Experimental Results

We compare HELMv2 to an additional setting, namely using CLIP in combination with the attention mechanism over token embeddings (HELMclip) on the RedBlueDoors and the TMaze environments. Fig. 4 shows the performance of HELMclip compared to the same methods as mentioned in Section 3. Again, HELMv2 consistenly outperforms HELMclip across both environments.

Next, we show additional results for the SHELM method which we introduced in Section 4. We instantiate SHELM with the RN50 backbone of CLIP and the corresponding RobProc mapping. Further, we add one more setting which substitutes the RobProc mapping of SHELM with a random orthogonal matrix sampled from the Haar distribution (Stewart, 1980) (SHELM+RandOrtho). We show results for SHELM, SHELM+RandOrtho, and HELMv2 in Fig. 5. Surprisingly, We find that SHELM+RandOrtho performs on-par with SHELM, and HELMv2. These results indicate that for solving our selected tasks, no semantic information is required. A possible reason for this is that CLIP is unable to sufficiently extract semantically meaningful features.

Finally, we conduct an additional analysis on how much the different methods preserve the semantics of natural images. In this regard, we perform the same quantitative analysis as in Appendix B for the different methods presented in this work. Again, we compute the MRR and NDCG for our

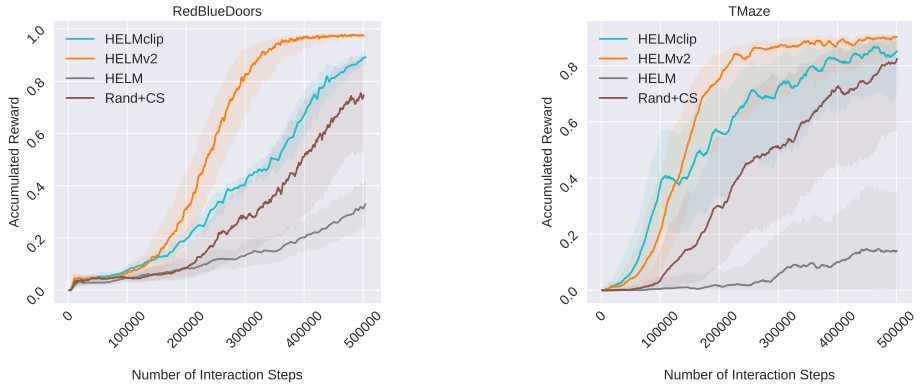


Figure 4: Mean IQM and 95% bootstrapped CIs across 30 seeds for RedBlueDoors (left), and TMaze (right) environments. HELMv2 consistently outperforms HELMclip on both environments.

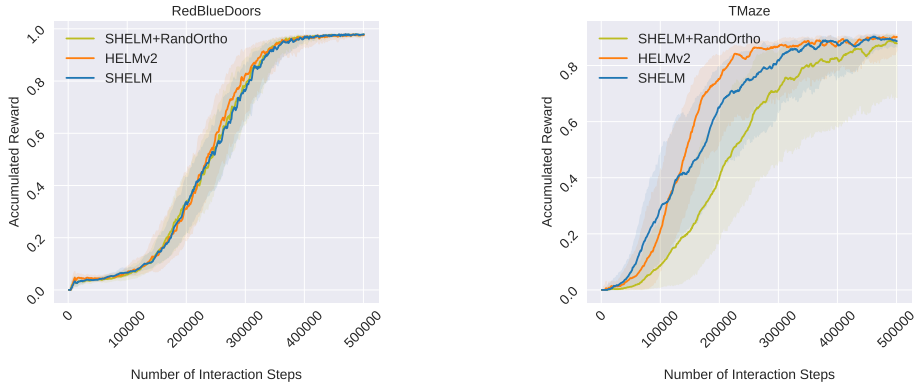


Figure 5: IQM and 95% bootstrapped CIs across 30 seeds on RedBlueDoors and TMaze environments. SHELM+RandOrtho performs on-par with HELMv2 and SHELM.

MSCOCO subset of image-caption pairs. Results are shown in Table 5. We observe that methods such as HELMv2, HELMclip, or SHELM+RandOrtho yield on-par or worse results compared to a random ranking. This is due to the fact, that for lower values of β the VocabAttn tends to output the mean over token embeddings. In this case, the resulting ranking is equal across image, and thus, results in lower MRR and NDCG. Also substituting the RobProc mapping with a random mapping drawn from the Haar distribution leads to random rankings. Contrary, SHELM yields a ranking that is closest the ranking performed by CLIP (see Table 3. Thus, for SHELM, we observe that most of the semantics are preserved. Adding VocabAttn after the RobProc mapping again results in loss of information and a suboptimal ranking.

	MRR (↑)	NDCG (↑)
Random	0.016±0.058	0.243±0.029
HELMclip, $\beta = 1$	9.1e-4±9.2e-4	0.221±0.023
HELMclip, $\beta = 10$	9.4e-4±9.5e-4	0.221±0.023
HELMclip, $\beta = 100$	0.017±0.061	0.242±0.034
HELMclip, $\beta = 1e3$	0.018±0.056	0.246±0.034
HELMclip, $\beta = 1e4$	0.018±0.056	0.246±0.035
HELMv2	0.018±0.057	0.244±0.031
SHELM+RandOrtho	0.019±0.068	0.245±0.033
SHELM+VocabAttn, $\beta = 1$	9.1e-4±9.2e-4	0.221±0.023
SHELM+VocabAttn, $\beta = 10$	0.001±0.001	0.222±0.023
SHELM+VocabAttn, $\beta = 100$	0.228±0.357	0.303±0.086
SHELM+VocabAttn, $\beta = 1e3$	0.217±0.368	0.297±0.083
SHELM+VocabAttn, $\beta = 1e4$	0.209±0.364	0.295±0.082
SHELM	**0.264±0.353**	**0.326±0.082**

Table 5: Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG) for tokens ranked in the LM embedding space for different ablation setups.