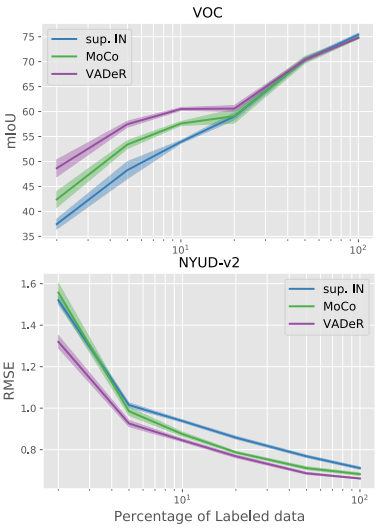
We thank the reviewers for the detailed and helpful reviews. We particularly acknowledge that reviewers find this work simple (R1, R2, R4), with strong empirical results (R1, R2, R4) and elegant and "completely novel" (R2). Next, we address the main concerns from reviewers.

Novelty (R1, R3). Current SSL methods (including MoCo, which we base upon) train only the bottom-up encoder w/o labels. They require pixelwise labels for the top-down decoder, which are initialized from scratch. As pointed by R2, the novelty of the approach is that we learn (unsupervised) pixel-level representations (ie, both the encoder and the decoder) instead of global representations (only the encoder). This allows for better initialization for dense labeling tasks and, to the best of our knowledge, has not been proposed before.

This is fundamentally different than an 'intensive data augmentation', as suggested by R3. By following R3's proposal, we would not be learning any features for the decoder (top-down path). The resulting representation would still encode the entire image and would not be particularly useful to pixel-level tasks. Even if we remove the pooling that proceeds the convolutional layers, the resulting downsampling is too aggressive for pixel-level tasks (factor of 32 in ResNet-50). Therefore, a decoder would still need to be initialized randomly.

Limited benefit when fine-tuning (R2, R3, R4). The benefits, however, increase when the amount of labeled data on fine-tuning stages is reduced. The figure on the right shows results (mean/std over 5 runs) when considering only fraction of the total data (2, 5, 10, 20, 50 and 100% of images), for both sem. seg. (VOC) and depth (NYUdepth). This result corroborates current research that show that SSL methods achieve better performance than supervised pre-training when the number of labeled data is limited.



Comparison with MoCo trained with 50 extra epochs (R2, R4). Training extra 50 epochs *with the original MoCo augmentations* does not make any statistical difference (on VOC and NYUDv2 fine-tuning) when compared with 200 epochs. This agrees with observations from MoCo-v2 paper (extra training helps much more on linear probing than on fine-tuning). When training the extra 50 epochs *with VADeR augmentations* (that is, same as MoCo, but discarding the pairs that does not share any pixel in common), the performance is slightly worse. This important discussion will be included on the revised version of the paper.

Training from scratch (R1, R4). It is not trivial to train randomly initialized networks with top-down path and skip-connections due to interaction between gradient of different paths. We found extremely challenging to train from randomly initialized network due to the very long training time for SSL, the set of hyperparameters and relatively limited resources. Popular methods (U-Net, SharpMask, FPN) start from an encoder initialized with supervised pre-training and random skip-connection/top-down weights. We follow the same approach, but initialize with MoCo weights (unsupervised) instead.

Missing citations (R1, R4). We will include the missing references and compare them with this work. (R1:) We will include the metric as in Doersch et al. However, comparison is not apples-to-apples since many training details are different (architecture, loss, etc). (R4:) Compared to Kanazawa et al., we believe the only similarity is the fact that correspondence flow is generated by applying transformation to images. Everything else is different, e.g., the high-level goal, the dataset (imagenet vs. fine-grained CUB), the loss, the evaluated tasks, the learning of features. We believe those works are actually complimentary, and WarpNet (that is initialized with ImageNet pretraining) could benefit from VADeR as other downstream tasks did.

Non-trivial engineering in augmentations (R1). We decided to use the same data augmentation as in MoCo-v2 (to facilitate comparison) with one small difference: making sure that at least 32 pixels belong to the two views (so that we can construct a correspondence map between the pixels). In early experiments we tried different minimum number of matching pixels (4, 8, 16, 32) per pair, and did not notice any qualitative difference.

Settings in the algorithm (R1). (i) We tried to use different pixels of the same image as negative examples in initial experiments (for proof of concept), but we could not make it work. Using pixels of other images is more natural (no need to try to find the ideal threshold distance) and fits naturally in the context of using a queue for negative samples. (ii) Momentum encoder might not be necessary, but we found easier to scale since SimCLR-like loss requires large batches to run (and VADeR requires even more memory due to the decoder). (iii) We observe very similar behavior w.r.t. to size of memory bank as reported in MoCo, going from 56.8 (with size 4096) mIoU in VOC to 58.3 (with size 65K).

Why not share params in f and g (L118)? (R3). Encoders f and g can or cannot share the same parameters. We follow one of the current SSL trends (InstDisc, CMC, PIRL, MoCo, etc.) and use different parameters for f and g. The latter is updated with momentum encoder, which allows for effective large number of negative samples.

Are the decoders fixed during training (L134)? (R3). No, both the encoder and decoder are trained. We mean "we place all the burden of representation learning on the network parameters (encoder-decoder)" (instead of compatibility function).

Results on classification for completeness (R4). We will report results of VADeR on linear probing for completeness on the revised version of the manuscript (although we expect it to be worse than other methods that learn global representations, as this is not our objective).