# Affective Retrofitted Word Embeddings

**Sapan Shah**[1,2]**, Sreedhar Reddy**[1]**, and Pushpak Bhattacharyya**[2]

[1]TCS Research, Tata Consultancy Services, Pune
[2]Indian Institute of Technology Bombay, Mumbai
{sapan.hs,sreedhar.reddy}@tcs.com
pb@cse.iitb.ac.in

## Abstract

Word embeddings learned using the distributional hypothesis (e.g., GloVe, Word2vec) do not capture the affective dimensions of valence, arousal, and dominance, which are present inherently in words. We present a novel retrofitting method for updating embeddings of words for their affective meaning. It learns a non-linear transformation function that maps pre-trained embeddings to an affective vector space, in a representation learning setting. We investigate word embeddings for their capacity to cluster emotion-bearing words. The affective embeddings learned by our method achieve better inter-cluster and intra-cluster distance for words having the same emotions, as evaluated through different cluster quality metrics. For the downstream tasks on sentiment analysis and sarcasm detection, simple classification models, *viz.* SVM and Attention Net, learned using our affective embeddings perform better than their pre-trained counterparts (more than 1.5% improvement in F1-score) and other benchmarks. Furthermore, the difference in performance is more pronounced in limited data setting.

## 1 Introduction

Affect refers to the experience of a feeling or emotion (Picard, 2000). This definition broadly encompasses sentiment, emotion, personality, and mood. Incorporating these affective aspects in text analysis can significantly benefit numerous NLP applications, including sentiment analysis, sarcasm detection, opinion mining, empathetic agents, etc. Words, being the smallest meaningful constructs in a language, have been the primary focus area for affect analysis in literature. The affective meaning of a word can be represented primarily using: (1) discrete affective labels such as joy, happiness, anger, etc., notable models include Plutchik's Wheel of Emotions (Plutchik, 1980), Ekman's model (Ekman, 1992), etc.; (2) dimensional models such as valence-arousal-dominance (VAD) model (Russell and Mehrabian, 1977), evaluation-potency-activity (EPA) model (Osgood et al., 1957), etc. that represent human affects in a continuous space. In this work, we focus on dimensional models since they capture more fine-grained information compared to the discrete models and are more expressive (Calvo and Mac Kim, 2013). The dimensional model in VAD represents a word and its affective meaning as a point in a 3-dimensional space that consists of valence (degree of pleasure or displeasure), arousal (degree of excitement or calmness), and dominance (degree of control or submission).

While pre-trained embeddings are good at capturing various lexico-semantic relations, do they encode the affective meaning of words? For example, consider *violate*, a word having low valence and high arousal. Table 1 shows the most similar words to *violate* as computed using cosine similarity with pre-trained Word2vec embeddings. This list includes words with high valence (e.g., *comply* and *obey*) as well as low arousal (e.g., *adhere*, *stipulate*), disregarding the affective meaning of *violate*. Similarly, *banish*, a word with low dominance, is one of the most similar words to *conquer*, a word having high dominance. This analysis suggests that the pre-trained word embeddings do not adequately encode the affective meaning of words.

It is well known in the community that the embeddings learned using the distributional hypothesis (Harris, 1954) mix semantic similarity with other types of semantic relatedness (Hill et al., 2015). For instance, though opposite in meaning, both *cheap* and *expensive* have similar embeddings since they occur in nearly identical contexts. This problem has been addressed by first borrowing semantic relations from knowledge sources such as WordNet, Paraphrase Database, etc., in the form of constraints and then using these constraints to learn joint specialization (Yu and Dredze, 2014; Liu et al., 2015) or retrofitting (Faruqui et al., 2015;

| word | Pre-trained Word2vec | VADProjWBal |
|---|---|---|
| violate (↓V;↑A) | contravene, violation, **abide**, prohibit, **adhere**, forbid, **comply**, contravention, **obey**, **stipulate** | contravene, prohibit, endanger, forbid, restrict, violation, oppose, **abide**, offend, discriminate |
| bombard (↓V;↑A) | barrage, overwhelm, saturate, zap, invade, terrorize, **ignore**, hurl, swarming, **scour** | overwhelm, terrorize, saturate, hurl, frighten, obliterate, gobble, zap, invade, unleash |
| conquer (↑A;↑D) | conquering, vanquish, overcome, liberate, annihilate, conquest, **banish**, unite, outwit, confront | conquering, vanquish, liberate, overcome, annihilate, unleash, unite, outwit, confront, wrest |

Table 1: Most similar words computed using cosine similarity: pre-trained Word2vec vs. embeddings retrofitted using our method (↑: high; ↓: low; V: Valence; A: Arousal; D: Dominance) - neighbours marked in bold do not agree with the probe word for affect dimensions

| **word** | **V** | **A** | **D** |
|---|---|---|---|
| adorable | 0.969 | 0.512 | 0.457 |
| suffering | 0.02 | 0.719 | 0.235 |
| conquer | 0.694 | 0.873 | 0.971 |
| slow | 0.357 | 0.073 | 0.131 |
| pretend | 0.49 | 0.528 | 0.542 |
| indulgence | 0.479 | 0.49 | 0.517 |

Table 2: Example words and their affect scores in the NRC VAD lexicon (**V:** Valence; **A:** Arousal; **D:** Dominance)

Mrkšić et al., 2016) models. However, these models focus mainly on synonymy, antonymy and hypernymy relations. Some recent efforts have used affective lexicons (Seyeditabari et al., 2019) or task-dependent distant supervision (Tang et al., 2016; Agrawal et al., 2018) to learn emotion embeddings. However, these methods rely only on discrete affective resources. Lately, a few attempts (Khosla et al., 2018; Chawla et al., 2019) have used resources created for dimensional models to learn affective embeddings. While the abovementioned approaches work well for some tasks, they do not generalize well across tasks and have not been evaluated extensively for affective aspects.

In this work, we present a simple yet effective retrofitting approach to learn VAD-enriched affective embeddings. For knowledge, it relies on the real-valued valence, arousal, and dominance scores available in the NRC VAD lexicon (Mohammad, 2018a). We hypothesize that when we map pre-trained embeddings to a vector space that is conducive to predicting VAD scores, the mapped vectors acquire affective meaning, resulting in affective embeddings. We design the mapping function as a non-linear transformation using a multi-layer feed-forward neural network. Given an input word, we first compute its affective embedding using the mapping function. The affective embedding is then linearly projected to a 3-dimensional vector space corresponding to the VAD dimensions. The scores present in the VAD lexicon are used to jointly learn both the mapping function as well as the linear VAD projection.

The affective embeddings learned using our method achieve better clustering for emotion bearing words. For downstream tasks on sentiment analysis and sarcasm detection, they perform better than their pre-trained counterparts and other benchmarks, with significant gains in limited data setting. The main contributions of this work are:

1. A simple yet effective approach to learn affective embeddings in a representation learning setting (Section 3).
2. A detailed evaluation showing better clustering achieved by our embeddings for emotion bearing words (Section 4.1).
3. A detailed evaluation on sentiment analysis and sarcasm detection showing the efficacy of our retrofitting method (Section 4.2).

## 2 NRC VAD Lexicon

Various lexical resources have been proposed in the literature to capture the affective meaning of words using dimensional models, e.g., ANEW (Bradley et al., 1999), Warriner's lexicon (Warriner et al., 2013), etc. In this work, we leverage the knowledge present in the VAD lexicon (Mohammad, 2018a) to learn affective embeddings. The lexicon provides real-valued scores in the range $[0, 1]$ for valence (**V**), arousal (**A**), and dominance (**D**) (0=low; 1=high) for more than 20,000 English words. Table 2 shows a few example words and their VAD scores. The word *adorable*, for instance, has high valence content with average arousal and dominance. We use the words in the lexicon and their

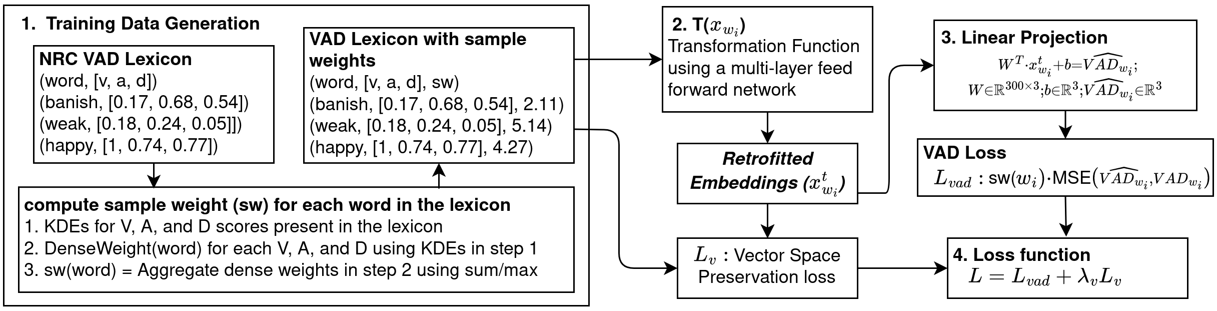
1. Training Data Generation
NRC VAD Lexicon
(word, [v, a, d])
(banish, [0.17, 0.68, 0.54])
(weak, [0.18, 0.24, 0.05]])
(happy, [1, 0.74, 0.77])
VAD Lexicon with sample weights
(word, [v, a, d], sw)
(banish, [0.17, 0.68, 0.54], 2.11)
(weak, [0.18, 0.24, 0.05], 5.14)
(happy, [1, 0.74, 0.77], 4.27)
compute sample weight (sw) for each word in the lexicon
1. KDEs for V, A, and D scores present in the lexicon
2. DenseWeight(word) for each V, A, and D using KDEs in step 1
3. sw(word) = Aggregate dense weights in step 2 using sum/max
2. T($x_{w_i}$)
Transformation Function using a multi-layer feed forward network
Retrofitted Embeddings ($x^t_{w_i}$)
$L_v$ : Vector Space Preservation loss
3. Linear Projection
$W^T \cdot x^t_{w_i} + b = \widehat{VAD}_{w_i}$;
$W \in \mathbb{R}^{300\times 3}; b \in \mathbb{R}^3; \widehat{VAD}_{w_i} \in \mathbb{R}^3$
VAD Loss
$L_{vad}$ : sw($w_i$)·MSE($\widehat{VAD}_{w_i}, VAD_{w_i}$)
4. Loss function
$L = L_{vad} + \lambda_v L_v$

Figure 1: Architecture for learning VAD-enriched affective retrofitted embeddings

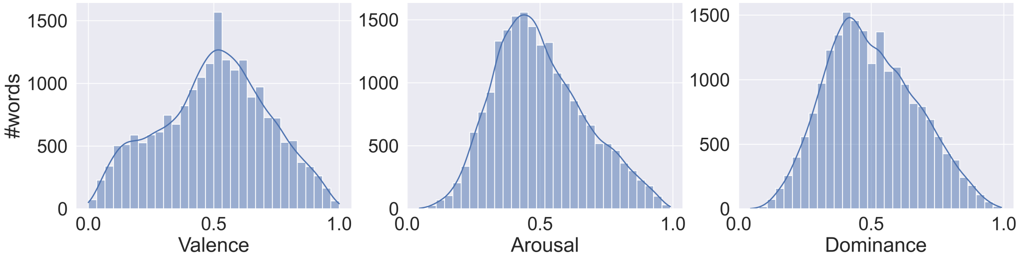

Figure 2: Histograms of valence, arousal, and dominance scores for the words in the VAD lexicon: #words with high/low affect scores are rare, whereas majority of words have average affect scores

affect scores as training data to learn our retrofitting model for affective embeddings.

## 3 Retrofitting method

Our goal is to learn a non-linear transformation function that maps pre-trained word embeddings to a vector space that encodes the affective meaning of words. The first question that arises here is: how do we measure or quantify the degree of affect content in a given vector space? We argue that it should be easy to extract the affective meaning of words from such a vector space. In fact, we hypothesize and show (refer Section 4) that a simple linear projection of word vectors from such a space to a 3-dimensional VAD space accurately extracts or predicts valence, arousal, and dominance scores of words. Therefore, we treat the *linear projection to the VAD space* as our objective criteria to learn the transformation function. To this end, the valence, arousal, and dominance scores present in the VAD lexicon provide the required training data. Figure 1 shows the overall architecture for learning our retrofitting model for affective embeddings.

**1. Training data generation:** A training example in our model consists of a word and its VAD scores. Generally, the number of words with high affect scores, either positive or negative, is limited in a language. Conversely, a large number of words have average affect scores. Figure 2 shows the histograms of VAD scores for the words in the VAD lexicon, depicting this language property. Regression models learned for target variables with such skewed distribution become biased, generally leading to better performance for common values than rare cases. However, the words that are referred more often to stress emotional or affective aspects in human communication generally have either positive or negative affect content as opposed to the average score. For example, consider words such as {happy, nightmare, weak, etc.}, and {indulgence, pretend, lease, etc.}. The former set contains words that exhibit affective aspects, whereas the latter contains words with minimal or no affective content. Since the words having extreme or rare VAD scores are of particular importance in our case, this imbalance in affect scores needs to be taken into account while learning our retrofitting model.

We employ a sample weighting approach with cost-sensitive learning to address the imbalanced regression problem described above. Specifically, sample weights are assigned to each word $w_i$ in the VAD lexicon such that the words with high/low affect scores get higher weights than those with average affect scores. We use the density-based weighting scheme (DenseWeight) proposed by Steininger et al. (2021) to compute sample weights. The following describes the process.

1. Apply kernel density estimator (KDE) to the valence scores of all words to obtain the density function $\mathrm{KDE}_\mathrm{v}$

2. Compute density $p_v(w_i)$ for each word $w_i$ using $\mathrm{KDE}_\mathrm{v}$

3. Apply the following weighting function to compute weights for all words

$$\mathrm{sw}_\mathrm{v}(w_i) = f_v(\alpha, w_i) = \max(1 - \alpha \cdot p(w_i), \epsilon)$$

Here, $\alpha \in [0, \inf)$ is a hyper-parameter. Setting it to 0 yields uniform weights. With increasing $\alpha$, sample weights of rare data points are emphasized more strongly. The parameter $\epsilon$ helps in avoiding negative or zero sample weights and is generally set to a small positive value, e.g., $5\mathrm{e}{-05}$. The process described above for valence is similarly applied for arousal and dominance to obtain $\mathrm{sw}_\mathrm{a}(w_i)$ and $\mathrm{sw}_\mathrm{d}(w_i)$, respectively. Finally, the sample weight $\mathrm{sw}(w_i)$ for the word $w_i$ is computed by aggregating these weights, i.e., $\mathrm{sw}(w_i) = \mathrm{aggregate}(\mathrm{sw}_\mathrm{v}(w_i), \mathrm{sw}_\mathrm{a}(w_i), \mathrm{sw}_\mathrm{d}(w_i))$. We experiment with two aggregation functions, i.e., max and sum.

**2. Transformation function:** We take the $d$-dimensional pre-trained embeddings of words as input and pass them through a non-linear transformation function to compute retrofitted embeddings, i.e., $x^t_{w_i} = \mathrm{T}(x_{w_i})$. This function is realized using a multi-layer feed-forward neural network with a corresponding set of network weights $N_T$.

**3. Linear projection to VAD space:** We linearly project the retrofitted embeddings $x^t_{w_i}$ to a 3-dimensional space that corresponds to valence, arousal and dominance dimensions, i.e., $\widehat{VAD_{w_i}} = W^T \cdot x^t_{w_i} + b$ where $W \in \mathbb{R}^{300\times3}; b \in \mathbb{R}^3$

**4. Loss function:** The VAD scores ($\widehat{VAD_{w_i}}$) predicted for the word $w_i$ using linear projection are compared to the corresponding VAD scores $VAD_{w_i}$, as present in the lexicon. We use mean squared error (MSE) as a loss function. As described earlier, we incorporate cost-sensitive learning to give higher sample weights to words having rare values for the affect scores. The sample weighted loss function used by our model is then,

$$L_{vad} = \textstyle\sum_{w_i} \mathrm{sw}(w_i) \cdot \mathrm{MSE}(\widehat{VAD_{w_i}}, VAD_{w_i})$$

It should be noted that the parameters for the linear projection ($W$ and $b$) as well as the transformation function ($N_T$) are learned jointly by our model. To obtain affective embeddings post training, we only require the transformation function, and the linear projection weights are discarded.

*Vector Space Preservation:* Pre-trained embeddings learned using the distributional hypothesis contain useful lexico-semantic relations. The transformation function learned by our model should preserve these relations while attending to the affective meaning of words. Similar to (Mrkšić et al., 2016; Glavaš and Vulić, 2018), we use a regularization term that penalizes transformations that drastically change the topology of pre-trained vector space. It measures the Euclidean distance between the pre-trained vector $x_{w_i}$ and its transformed version $\mathrm{T}(x_{w_i})$, i.e., $L_v = \sum_{w_i} \|x_{w_i} - \mathrm{T}(x_{w_i})\|_2$. The final loss function used by our model is then,

$$L = L_{vad} + \lambda_v L_v \tag{1}$$

where $\lambda_v$ is a hyper-parameter that controls how strictly the topology of the original vector space is preserved. The loss function also includes L2-regularization for the parameters $N_T$, $W$, and $b$.

## 4 Experimental Results

To evaluate our method, we experimented with 300-dimensional pre-trained embeddings in Word2vec[1] (Mikolov et al., 2013) and GloVe[2] (Pennington et al., 2014). Due to space constraints, we discuss only Word2vec results here (refer Appendix B for GloVe). The complete hyper-parameter grid search details, computational cost, etc. are detailed in Appendix A. As discussed earlier, the transformation function that maps pre-trained word embeddings to an affective vector space is learned in a regression setting using the loss function in Eq. 1. This loss function contains two contrasting terms, *viz.* VAD regression loss ($L_{vad}$) and vector space preservation loss ($L_v$). The hyper-parameter $\lambda_v$ provides a knob to balance these contrasting terms and needs to be set at the right value to learn a meaningful transformation function. Setting a very high value for $\lambda_v$ will make our model ignore the affective content of words, thereby learning retrofitted embeddings nearly identical to their pre-trained version. Conversely, a low value of $\lambda_v$ may produce embeddings that predominantly contain affective meaning at the expense of forgetting lexico-semantic rela-

[1] https://code.google.com/archive/p/word2vec/

[2] https://nlp.stanford.edu/data/glove.42B.300d.zip

tions present in the pre-trained vector space, possibly leading to degraded performance on end-tasks.

To select the best hyper-parameter configuration, we conduct two experiments. (1) We directly select the configuration that gives the least MSE[3] in predicting VAD scores (referred as **VADProjW**) (2) We first compute the mean cosine distance between the pre-trained and affective embeddings of words and select configurations with a distance $< 0.15$. We then choose the best configuration (with the least MSE in VAD prediction) amongst the filtered list (referred as **VADProjWBal**).

### Quantifying affective content

Our primary objective is to incorporate affective meaning into pre-trained embeddings. A few relevant questions in this context are: how much affective content do pre-trained embeddings have? Does our retrofitting method improve it? As discussed earlier, it should be easy to extract VAD scores if the vector space is sensitive to affective aspects. In other words, a simple linear combination of values present in the embeddings vector shall predict the VAD scores with reasonable accuracy. To investigate this, we built a linear regression model for predicting VAD scores using the VAD lexicon dataset. With pre-trained Word2vec, the model achieved an MSE of $0.0345$. On the other hand, the affective embeddings in VADProjWBal resulted in an MSE of $0.0157$, about 55% reduction in error (25% with affective GloVe embeddings). These results indicate that the retrofitted vector space learned by our method is sensitive to the affective meaning of words. Indeed, the neighbours computed using VADProjWBal embeddings are affect-aware, as evident from the exemplar words in Table 1.

### Compared work

The retrofitting approaches proposed in the literature employ two types of constraints: *attract* constraints that pull similar (e.g., synonyms, hypernyms, etc.) words together, and *repel* constraints that push non-similar (e.g., antonyms) word pairs away from each other. **Counterfit** (Mrkšić et al., 2016) uses a loss function that brings attract pairs closer and pushes repel pairs apart. However, it updates embeddings of words present in attract and repel constraints in isolation without considering their relations to other words. To address this, **Attract-Repel (AR)** (Mrkšić et al., 2017) performs context-sensitive vector updates using a hinge loss function that additionally considers in-batch negative example words. Both the Counterfit and AR methods retrofit vectors of only those words that are present in the constraints (*seen words*). The embeddings for all other words are not updated. Post-specialization methods use a mapping function that takes embeddings of seen words as input to learn a non-linear transformation and then uses it to retrofit unseen words. The approach proposed by Ponti et al. (2018) uses a generative adversarial network to learn the mapping function (**AR+PS**), with AR to retrofit seen words.

The methods described above use general purpose resources for updating pre-trained embeddings. We also compare our work with methods that use resources created for discrete or dimensional models of affect. Agrawal et al. (2018) (**EWE**) use distant supervision to create emotion labelled data and then apply a recurrent neural network to learn emotion embeddings. The embeddings (**EEArmin**) proposed by Seyeditabari et al. (2019), on the other hand, employ the counterfit method directly on *(word, emotion)* pairs. Both these approaches use NRC EmoLex (Mohammad and Turney, 2013), a resource that provides discrete emotion labels. Khosla et al. (2018) propose 303-dimensional affective embeddings (**Aff2vec**) by appending valence, arousal, and dominance scores of words to their counterfitted embeddings. The embeddings in **SentiEmbs** (Yu et al., 2017) are refined to incorporate sentiment information using valence scores in the Warriner's lexicon.

In addition to retrofitting, we also compare our method with two joint learning approaches. Semantic word embeddings (**SWE**) developed by Liu et al. (2015) directly integrate constraints from WordNet into the optimization objective of Word2vec. Chawla et al. (2019) (**JointAff2vec**) first generate constraints by combining relations in WordNet with the affect scores in Warriner's lexicon. These constraints are then used as part of the cost function of pre-trained embedding models.

We use pre-trained embeddings as a baseline. Additionally, we concatenate the embeddings of words with their valence, arousal, and dominance scores to create an affect-aware baseline (referred as **Word2vec$\oplus$VAD**, 303-dimensional vectors).

[3] computed using 10% words set aside as a validation set

| **Embeddings** | **ARI**↑ | **FMS**↑ | **AMIS**↑ | **V-measure**↑ | **VDist**↓ | **RankAvg**↓ |
|---|---|---|---|---|---|---|
| Word2vec | 0.0492(9) | 0.1849(9) | 0.075(9) | 0.0768(9) | 0(1) | 5 |
| Word2vec⊕VAD | 0.0995(4) | 0.229(4) | 0.1417(8) | 0.1434(8) | NA(7) | 6.5 |
| Counterfit | 0.0762(8) | 0.1814(10) | 0.1495(7) | 0.1518(7) | 0.1803(4) | 6 |
| AR | 0.0794(7) | 0.186(8) | 0.1538(5) | 0.1561(5) | 0.2556(5) | 5.63 |
| AR+PS | 0.0913(6) | 0.2051(6) | 0.159(3) | 0.1613(3) | 0.1326(3) | 3.75 |
| SWE† | 0.0215(10) | 0.1713(11) | 0.044(10) | 0.0459(10) | 0.9903(10) | 10.13 |
| Aff2vec | 0.0914(5) | 0.1978(7) | 0.1567(4) | 0.1591(4) | NA(7) | 6 |
| EEArmin† | 0.3655(1) | 0.4468(1) | 0.5495(1) | 0.5507(1) | 0.9986(11) | 6 |
| SentiEmbs† | 0.0007(11) | 0.3000(2) | 0.0085(11) | 0.0126(11) | 0.4382(9) | 8.89 |
| VADProjW | 0.1237(2) | 0.2466(3) | 0.1842(2) | 0.1858(2) | 0.3461(6) | 4.13 |
| VADProjWBal | 0.1036(3) | 0.2288(5) | 0.1529(6) | 0.1546(6) | 0.1006(2) | **3.5** |

Table 3: External cluster validity indices with pre-trained Word2vec and its updated versions, our method in last two rows - [↓: lower values are better; ↑: higher values are better] - The value in bracket specifies the rank of a given embedding for the metric (lower ranks are better); The embeddings marked with † may not perform well on affective end-tasks since they change the topology of pre-trained vector space drastically (very high VDist)

### 4.1 Clustering of Emotion-bearing Words

The primary objective of our retrofitting method is to incorporate the affective meaning of words into pre-trained embeddings. In this context, it is natural to ask, do the affective embeddings learned by our method also reliably capture emotion aspects? One way to quantify this is to check whether the learned embeddings are similar for words that exhibit the same emotion. Alternatively, are words having the same emotion clustered together in the vector space? To study this, we use NRC EmoLex (Mohammad and Turney, 2013), a lexicon that provides English words and their associations with Plutchik's eight basic emotion categories. A few example (`word`, *emotion*) pairs present in the lexicon include (`adorable`, *joy*), (`suffering`, *fear*), and so on. We cluster all the words present in EmoLex using K-means (#means k=8) algorithm, which uses the embeddings of words as input features. Since the true emotion category labels are available, we apply various external cluster validity indices (refer to Scikit-learn user guide) such as adjusted rand index (ARI), Fowlkes Mallows score (FMS), adjusted mutual information score (AMIS) and V-measure, to quantify clustering quality. In addition to good clustering, affective embeddings shall also preserve the topology of pre-trained vector space. To measure this, we compute the average cosine distance between pre-trained and affective embeddings for words in EmoLex (referred as **VDist**).

The pre-trained Word2vec embeddings perform poorly across all clustering indices, as shown in Table 3. This result indicates that they do not consider the emotion aspects of words. The pre-trained embeddings, when made affect-aware using a simple concatenation with the VAD scores (Word2vec⊕VAD baseline), perform significantly better. However, vector distances perturbed due to the extra 3-dimensions may adversely impact other useful semantic relations captured originally by the distributional hypothesis. The embeddings from past retrofitting methods (Counterfit, AR, and AR+PS) that use general resources, reasonably improve clustering beyond the pre-trained baseline. However, their (except for AR+PS) VDist is high, suggesting that they did not maintain semantic relations present in Word2vec. The embeddings produced by the joint learning approach in SWE perform poorly on both the clustering and vector space preservation metrics. The EEArmin embeddings have completely overfitted for clustering, with extremely poor VDist. On the other hand, the EWE embeddings[4] have poor clustering quality as they are nearly identical to their pre-trained version (VDist=0.0085). The embeddings in SentiEmbs are optimized only for coarse-grained sentiments, possibly leading to poor clustering on fine-grained emotions. Although Aff2vec embeddings achieve reasonably good clustering, similar to Word2vec⊕VAD, we cannot measure their VDist due to the extra 3-dimensions. VADProjW embeddings, selected based only on VAD prediction accuracy, achieve substantially good clustering but have poor VDist, as expected. The affective

[4] EWE applicable only for GloVe (refer Appendix B); embeddings not available for JointAff2vec

| Task | Dataset | #class | Size | #token | Type | Vocab | Source |
|---|---|---|---|---|---|---|---|
| Sentiment analysis | SST2 | 2 | 9,613 | 162,783 | sentence | 17,630$_1$ | (Socher et al., 2013) |
| | SST5 | 5 | 11,855 | 199,120 | sentence | 19,631$_1$ | (Socher et al., 2013) |
| | SemEval | 3 | 61,854 | 1,174,626 | tweet | 23,005$_2$ | (Rosenthal et al., 2017) |
| Sarcasm detection | Mustard++ | 2 | 1,202 | 14,219 | utterance | 2,632$_1$ | (Ray et al., 2022) |

Table 4: Dataset statistics for affective end-tasks (subscript in **Vocab** indicate minimum frequency threshold)

| Embeddings | SVM | | | | AttnNet | | | |
|---|---|---|---|---|---|---|---|---|
| | SST2 | SST5 | SemEval | Mus++ | SST2 | SST5 | SemEval | Mus++ |
| Word2vec | 0.8155 | 0.4249 | 0.6203 | 0.5565 | 0.8012 | 0.4036 | 0.6347 | 0.5208 |
| Word2vec⊕VAD | 0.816 | **0.4385** | 0.6369 | 0.5481 | 0.7957 | 0.3584 | **0.6374** | **0.5583** |
| Counterfit | 0.8122 | 0.4271 | 0.6294 | 0.569 | 0.7315 | 0.3683 | 0.6303 | 0.4667 |
| AR | 0.8133 | 0.3946 | 0.5947 | 0.5607 | 0.7738 | 0.3869 | 0.6289 | 0.5125 |
| AR+PS | 0.8149 | 0.4167 | 0.6007 | 0.5272 | 0.7952 | **0.4109** | 0.6283 | 0.5292 |
| SWE | 0.7304 | 0.3593 | 0.555 | 0.4979 | 0.6524 | 0.3054 | 0.5634 | 0.5167 |
| Aff2vec | **0.8166** | 0.407 | 0.6119 | 0.5439 | 0.7814 | 0.4036 | 0.629 | 0.5458 |
| EEArmin | 0.771 | 0.3887 | 0.5964 | **0.5732** | 0.7529 | 0.3751 | 0.6191 | 0.5167 |
| SentiEmbs | 0.7551 | 0.3647 | 0.5726 | 0.569 | 0.7057 | 0.3394 | 0.5529 | **0.5583** |
| JointAff2vec* | 0.7534 | 0.405 | - | - | - | - | - | - |
| VADProjW | 0.8089 | 0.419 | **0.6402** | **0.5858** | **0.8144** | 0.3819 | 0.6373 | 0.525 |
| VADProjWBal | **0.8204** | **0.4425** | **0.6411** | 0.5649 | **0.8105** | **0.429** | **0.6379** | **0.5667** |

Table 5: Micro F1-scores for SVM and AttnNet with various embeddings as input: Experiments with Word2vec as baseline (**Bold+Underline**: highest; **Bold**: next highest) (*JointAff2vec: Chawla et al. (2019) report results only for SST2 and SST5; **EWE embeddings applicable only for GloVe, not available for Word2vec)

embeddings in VADProjWBal provide the right balance overall with substantially good clustering along with a low value for VDist.

In addition to scores, Table 3 also reports the rank (mentioned in bracket) of various embeddings for each metric. The weighted average[5] (RankAvg in Table 3) computed across metrics suggests that VADProjWBal achieves the best performance overall, closely followed by AR+PS embeddings.

## 4.2 Evaluation on Downstream Tasks

We evaluate our method on two affective end-tasks: (1) Sentiment analysis on Stanford sentiment treebank with both the binary (SST2) and graded (SST5) variants and SemEval 2017 task 4A containing tweet messages; (2) Sarcasm detection using Mustard++ dataset that contains sitcom utterances. Table 4 details the statistics of these datasets. We use a probing framework, similar to (Agrawal et al., 2018), to evaluate embeddings on downstream tasks. Specifically, we use two classification models: support vector machine (SVM), and attention network (AttnNet). The input features for SVM are computed by averaging the embeddings of tokens present in a given sentence/tweet/utterance. Whereas the token embeddings, as a sequence, are passed as input to an attention layer followed by softmax to compute cross-entropy loss for AttnNet.

Table 5 reports the micro F1-scores for SVM and AttnNet. The pre-trained Word2vec seems to be a strong baseline to beat on both the tasks. Using VAD scores explicitly as input makes Word2vec⊕VAD an even stronger baseline, illustrating the role affect dimensions play, especially for affective downstream tasks. Both retrofitting (Counterfit, AR, AR+PS) and joint specialization (SWE) methods have been shown to improve tasks such as dialogue state tracking, text simplification, etc. However, for the affective tasks, they could not even beat the baselines. This is probably because these methods focus only on relations such as synonymy, antonymy, and hypernymy that are present in general resources and are not tailored for affec-

[5] both clustering metrics and VDist are given equal weights, i.e., 0.25 for each clustering metric and 1 for VDist; In VDist, the mean score across all methods is used to arrive at ranks for 'NA'

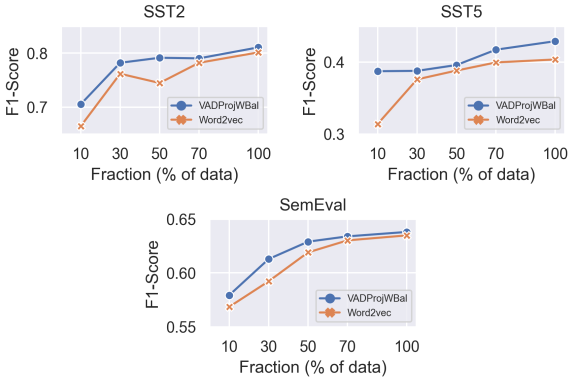

Figure 3: Data size vs. micro F1-score for pre-trained Word2vec and VADProjWBal in limited data setting

tive dimensions of meaning. Though both Aff2vec and EEArmin embeddings are retrofitted using affective resources, they could not beat baseline embeddings, possibly due to the drastic changes they allow to the topology of pre-trained vector space (high VDist). JointAff2vec embeddings, obtained by the joint learning approach using both affect resource and WordNet, could not perform well. This finding coincides with the observation in (Mrkšić et al., 2017) that joint learning approaches generally have lower performance compared to retrofitting methods. The lower value of VDist (0.009) suggests that the EWE embeddings are nearly identical to their pre-trained version having no capacity to improve beyond the baseline. Though optimized for sentiments, SentiEmbs could not perform well even on the sentiment analysis task. Overall, VADProjWBal, the embeddings retrofitted by our method to respect affective meaning while also being considerate to the topology of input vector space, achieve the highest F1-score for both SVM and AttnNet on sentiment analysis task. On sarcasm detection, they perform better than both the baselines and achieve the highest F1-score with AttnNet.

### 4.2.1 Limited Data Experiments

We further evaluate embeddings for their performance in a low resource setting. From the sentiment analysis datasets, we first sample sub-datasets of various sizes, such as 10%, 30%, etc., and then compare the F1-score of pre-trained Word2vec with VADProjWBal across the data sizes. As evident from Figure 3, VADProjWBal significantly outperforms pre-trained Word2vec in a low data regime. The difference in performance decreases gradually with an increase in dataset size. This result points to the fact that the knowledge of the affective meaning of words as captured by our method helps improve end tasks, especially in a limited data scenario.

## 5 Related Work

Word embeddings built using the distributional hypothesis have been studied extensively in the literature for the types of semantic relations they encode. It has been observed that they mix semantic similarity with other types of relatedness (Hill et al., 2015), potentially leading to degraded end-task performance. Various joint learning (Yu and Dredze, 2014; Liu et al., 2015) or retrofitting (Faruqui et al., 2015; Mrkšić et al., 2016; Shah et al., 2020) models address this problem by leveraging semantic relations from resources such as WordNet, Paraphrase Database, etc. However, they focus mainly on synonymy, antonymy, and hypernymy relations. To inject affective meaning into word embeddings, a few attempts (Agrawal et al., 2018; Seyeditabari et al., 2019) have recently used resources such as EmoLex (Mohammad and Turney, 2013) and affect intensity lexicon (Mohammad, 2018b) that cater to discrete affective models. These methods, however, are limited by the coarse-grained affect labelling and lack finer affective interpretations. Lately, Khosla et al. (2018) and Chawla et al. (2019) have used dimensional model resources such as Warriner's lexicon (Warriner et al., 2013) and VAD lexicon (Mohammad, 2018a) to encode fine-grained affective meaning.

Different from affect, there also exist lexicons that can be used to ground the semantic meaning of affect bearing words into other modalities. For example, colors in the NRC word-color association (e.g. `danger` - *red*) lexicon (Mohammad, 2011); perceptual modalities and action effectors in Lancaster sensorimotor norms (Lynott et al., 2019); robot state behavior (Moro et al., 2020), etc.

A large body of work focuses on learning task-specific affective embeddings. These methods first generate a noisy labelled dataset using distant supervision and then use it to update word embeddings or learn them from scratch. Notable works include sentiment-aware embeddings (Tang et al., 2014, 2016) using tweet data, affective embeddings (Felbo et al., 2017) using tweet emojis, emotion-enriched embeddings (Agrawal et al., 2018) using product reviews, etc. However, the embeddings learned from these methods are customized with dataset-specific nuances and might also model noise inherently present due to distant supervision. Due to this, they do not generalize well across other related tasks.

The affective embeddings learned by our retrofitting method are not only accurate compared to the methods described above, as evident from the clustering experiments, but also work well on the related affective end-tasks.

## 6 Summary and Future Work

We present a simple yet effective retrofitting method to learn affective embeddings using the NRC VAD lexicon. The affect scores in the lexicon are used as training data to learn a transformation function in a representation learning setting that maps pre-trained embeddings to an affective vector space. The embeddings learned by our method perform better than their pre-trained version and other benchmarks in both the intrinsic task of clustering emotion-bearing words and the affective downstream tasks in sentiment analysis and sarcasm detection. We are currently extending our retrofitting approach to other affective resources such as affect intensity lexicon (Mohammad, 2018b) and EmoLex (Mohammad and Turney, 2013). We also plan to develop a similar approach for contextualized word embeddings.

## References

Ameeta Agrawal, Aijun An, and Manos Papagelis. 2018. Learning emotion-enriched word representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 950–961, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Margaret M. Bradley, Peter J. Lang, Margaret M. Bradley, and Peter J. Lang. 1999. Affective norms for english words (anew): Instruction manual and affective ratings. The Center for Research in Psychophysiology, University of Florida.

Rafael A. Calvo and Sunghwan Mac Kim. 2013. Emotions in text: Dimensional and categorical models. *Computational Intelligence*, 29(3):527–543.

Kushal Chawla, Sopan Khosla, Niyati Chhaya, and Kokil Jaidka. 2019. Pre-trained affective word representations. In *2019 8th International Conference on Affective Computing and Intelligent Interaction (ACII)*, pages 1–7.

Paul Ekman. 1992. An argument for basic emotions. *Cognition and Emotion*, 6:169–200.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, Colorado. Association for Computational Linguistics.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

Goran Glavaš and Ivan Vulić. 2018. Explicit retrofitting of distributional word vectors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 34–45, Melbourne, Australia. Association for Computational Linguistics.

Zellig Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

Felix Hill, Roi Reichart, and Anna Korhonen. 2015. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Sopan Khosla, Niyati Chhaya, and Kushal Chawla. 2018. Aff2Vec: Affect–enriched distributional word representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2204–2218, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. Cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Quan Liu, Hui Jiang, Si Wei, Zhen-Hua Ling, and Yu Hu. 2015. Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1501–1511, Beijing, China. Association for Computational Linguistics.

Dermot Lynott, Louise Connell, Marc Brysbaert, James Brand, and James Carney. 2019. The lancaster sensorimotor norms: multidimensional measures of perceptual and action strength for 40,000 english words. *Behavior Research Methods*, 52:1271 – 1291.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Saif Mohammad. 2011. Even the abstract have color: Consensus in word-colour associations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–373, Portland, Oregon, USA. Association for Computational Linguistics.

Saif M. Mohammad. 2018a. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words. In *Proceedings of The Annual Conference of the Association for Computational Linguistics (ACL)*, Melbourne, Australia.

Saif M. Mohammad. 2018b. Word affect intensities. In *Proceedings of the 11th Edition of the Language Resources and Evaluation Conference (LREC-2018)*, Miyazaki, Japan.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. *Computational Intelligence*, 29(3):436–465.

Daniele Moro, Gerardo Caracas, David McNeill, and Casey Kennington. 2020. Semantics with feeling: Emotions for abstract embedding, affect for concrete grounding. In *Proceedings of the 24th Workshop on the Semantics and Pragmatics of Dialogue - Full Papers*, Virtually at Brandeis, Waltham, New Jersey. SEMDIAL.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148, San Diego, California. Association for Computational Linguistics.

Nikola Mrkšić, Ivan Vulić, Diarmuid Ó Séaghdha, Ira Leviant, Roi Reichart, Milica Gašić, Anna Korhonen, and Steve Young. 2017. Semantic specialisation of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*, 5:309–324.

C.E. Osgood, G.J. Suci, and P.H. Tenenbaum. 1957. *The Measurement of meaning*. University of Illinois Press, Urbana:.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Rosalind W. Picard. 2000. *Affective Computing*. The MIT Press.

Robert Plutchik. 1980. A general psychoevolutionary theory of emotion. *Theories of emotion*.

Edoardo Maria Ponti, Ivan Vulić, Goran Glavaš, Nikola Mrkšić, and Anna Korhonen. 2018. Adversarial propagation and zero-shot cross-lingual transfer of word vector specialization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 282–293, Brussels, Belgium. Association for Computational Linguistics.

Anupama Ray, Apoorva Nunna, and Pushpak Bhattacharyya. 2022. A multimodal corpus for emotion recognition in sarcasm. In *Proceedings of the 13th Edition of the Language Resources and Evaluation Conference (LREC-2022)*, Marseille, France.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada. Association for Computational Linguistics.

James A Russell and Albert Mehrabian. 1977. Evidence for a three-factor theory of emotions. *Journal of Research in Personality*, 11(3):273–294.

Scikit-learn user guide. Clustering performance evaluation. Online; accessed 01-February-2022.

Armin Seyeditabari, Narges Tabari, Shafie Gholizadeh, and Wlodek Zadrozny. 2019. Emotional embeddings: Refining word embeddings to capture emotional content of words. *ArXiv*, abs/1906.00112.

Sapan Shah, Sreedhar Reddy, and Pushpak Bhattacharyya. 2020. A retrofitting model for incorporating semantic relations into word embeddings. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1292–1298, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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.

Michael Steininger, Konstantin Kobs, Padraig Davidson, Anna Krause, and Andreas Hotho. 2021. Density-based weighting for imbalanced regression. *Mach. Learn.*, 110(8):2187–2211.

Duyu Tang, Furu Wei, Bing Qin, Nan Yang, Ting Liu, and Ming Zhou. 2016. Sentiment embeddings with applications to sentiment analysis. *IEEE Transactions on Knowledge and Data Engineering*, 28(2):496–509.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for Twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1555–1565, Baltimore, Maryland. Association for Computational Linguistics.

Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, 45:1191–1207.

Liang-Chih Yu, Jin Wang, K. Robert Lai, and Xuejie Zhang. 2017. Refining word embeddings for sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 534–539, Copenhagen, Denmark. Association for Computational Linguistics.

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 545–550, Baltimore, Maryland. Association for Computational Linguistics.

# A Training details

This section details the hyper-parameters and the best combinations selected thereof. The transformation function T in our retrofitting method is implemented using a multi-layer feed-forward neural network. The corresponding hyper-parameters are:- number of hidden layers: $\{1, 2, 3\}$, size of hidden layer: $\{200, 300\}$, activations: $\mathrm{LeakyReLU}$, dropout: $0.5$, and L2 regularization: $1\mathrm{e}{-5}$. We use Adam (Kingma and Ba, 2014) optimization algorithm with batch size $128$, number of epochs $200$, and a learning rate of $0.001$. The learning rate is reduced on a plateau (patience=5) with a factor of $0.2$, with a minimum learning rate set to $1\mathrm{e}{-6}$. We computed sample weights for the words in the VAD lexicon with the $\alpha$ parameter in the weighting function set to $\{0.75, 1, 1.1, 1.25, 1.5\}$. We finally used sample weights obtained for $\alpha = 1.25$ since the corresponding weights seem to provide a good balance between rare and common words. We use $\max$ as the aggregation function to combine sample weights for valence, arousal, and dominance. The hyper-parameter $\lambda_v$ is varied from $0.01$ to $0.05$ with a step size of $0.01$ and from $0.1$ to $1$ with a step size of $0.2$. We set aside $10\%$ words in the VAD lexicon for validation. For experimentation, we used CPU machines with 64GB RAM and 20 core CPUs. Each configuration, on average, took about 20 minutes to run.

For both Word2vec and GloVe, we conduct experiments with two configurations to generate retrofitted embeddings. One configuration is selected only on the basis of VAD prediction quality (the configuration with the least MSE on the validation set). The second configuration considers vector space preservation in addition to the VAD prediction quality. Table 6 reports these configurations.

# B Experimental results for GloVe

Table 7 reports clustering experiments for GloVe pre-trained baseline, the corresponding affective embeddings, and other benchmarks. Table 8 reports results for sentiment analysis and sarcasm detection tasks for SVM and Attention network with GloVe as the base embeddings.

| **hyperparameter** | **Word2vec** | | **GloVe** | |
|---|---|---|---|---|
| | VADProjWBal | VADProjW | VADProjGBal | VADProjG |
| #layers | 1 | 2 | 1 | 2 |
| #hidden units | 300 | 300 | 300 | 200 |
| activation | LReLU | LReLU | LReLU | LReLU |
| dropout | 0.5 | 0.5 | 0.5 | 0.5 |
| L2-regularization | $1e{-5}$ | $1e{-5}$ | $1e{-5}$ | $1e{-5}$ |
| batch-size | 128 | 128 | 128 | 128 |
| learning rate | 0.001 | 0.001 | 0.001 | 0.001 |
| $\alpha$ | 1.25 | 1.25 | 1.25 | 1.25 |
| $\lambda_v$ | 0.03 | 0.01 | 0.02 | 0.01 |

Table 6: Selected hyper-parameter configurations for affective retrofitted embeddings (1) Word2vec:- VADProjW has the least MSE for VAD prediction; VADProjWBal additionally has VDist < 0.15 (2) GloVe:- VADProjG has the least MSE for VAD prediction; VADProjGBal additionally has VDist < 0.15

| **Embeddings** | **ARI**↑ | **FMS**↑ | **AMIS**↑ | **V-measure**↑ | **VDist**↓ | **RankAvg**↓ |
|---|---|---|---|---|---|---|
| GloVe | 0.0408(10) | 0.1764(11) | 0.0731(10) | 0.0749(10) | 0(1) | 5.63 |
| GloVe⊕VAD | 0.0482(9) | 0.1818(9) | 0.0898(9) | 0.0915(9) | NA(7) | 8 |
| Counterfit | 0.0897(4) | 0.1969(5) | 0.1634(3) | 0.1657(3) | 0.1740(6) | 4.89 |
| AR | 0.0749(7) | 0.1802(10) | 0.1479(7) | 0.1502(7) | 0.0977(3) | 5.38 |
| AR+PS | 0.0853(5) | 0.1911(7) | 0.1607(4) | 0.1630(4) | 0.1257(5) | 5 |
| EWE | 0.0602(8) | 0.1924(6) | 0.1071(8) | 0.1089(8) | 0.0085(2) | 4.75 |
| Aff2vec | 0.0824(6) | 0.1877(8) | 0.1574(5) | 0.1598(5) | NA(7) | 6.5 |
| EEArmin† | 0.3764(1) | 0.4566(1) | 0.5501(1) | 0.5514(1) | 1.0152(11) | 6 |
| SentiEmbs† | 0.0009(11) | 0.2974(2) | 0.0135(11) | 0.0176(11) | 0.4329(10) | 9.38 |
| VADProjG | 0.106(2) | 0.2278(3) | 0.1658(2) | 0.1674(2) | 0.3247(9) | 5.63 |
| VADProjGBal | 0.0976(3) | 0.2203(4) | 0.1543(6) | 0.1559(6) | 0.1029(4) | **4.38** |

Table 7: External cluster validity indices (with k=8) for pre-trained GloVe and its retrofitted versions (↓: lower values are better; ↑: higher values are better) - The value in bracket specifies the rank of a given embedding for the metric (lower ranks are better); RankAvg is a weighted average of ranks across metrics (equal weights considered for both the clustering metrics and VDist, i.e., 0.25 for each clustering metric and 1 for VDist); The embeddings marked with † may not perform well on affective end-tasks since they change the topology of pre-trained vector space drastically (very high VDist)

| **Embeddings** | **SVM** | | | | **AttnNet** | | | |
|---|---|---|---|---|---|---|---|---|
| | **SST2** | **SST5** | **SemEval** | **Mus++** | **SST2** | **SST5** | **SemEval** | **Mus++** |
| GloVe | 0.8034 | 0.4122 | 0.6131 | 0.5333 | 0.782 | 0.4176 | 0.637 | 0.5458 |
| GloVe⊕VAD | 0.8029 | 0.4136 | 0.615 | 0.5333 | 0.7919 | 0.4253 | 0.6322 | 0.5 |
| Counterfit | 0.8007 | **0.4181** | 0.624 | 0.5105 | 0.7798 | 0.3855 | 0.6261 | **0.575** |
| AR | 0.8051 | 0.3932 | 0.5755 | 0.5063 | 0.7381 | 0.357 | 0.6381 | 0.5333 |
| AR+PS | **0.8078** | 0.4036 | 0.601 | 0.4979 | 0.743 | 0.4235 | 0.6276 | 0.525 |
| EWE | 0.7974 | 0.402 | 0.6049 | 0.5523 | 0.7727 | 0.3701 | 0.6182 | 0.4708 |
| Aff2vec | 0.7831 | 0.3893 | 0.5725 | 0.523 | 0.7655 | 0.4 | 0.6259 | 0.5125 |
| EEArmin | 0.7644 | 0.3805 | 0.5604 | 0.5397 | 0.7282 | 0.3561 | 0.6176 | **0.5792** |
| SentiEmbs | 0.7397 | 0.3633 | 0.5511 | 0.5356 | 0.67 | 0.3326 | 0.5418 | 0.475 |
| JointAff2vec* | 0.8035 | 0.4145 | - | - | - | - | - | - |
| VADProjG | 0.8012 | 0.4149 | **0.6356** | **0.5625** | **0.7957** | **0.4244** | **0.6415** | 0.525 |
| VADProjGBal | **0.8083** | **0.4267** | **0.6414** | **0.5708** | **0.804** | **0.4262** | **0.6405** | 0.55 |

Table 8: Micro F1-scores for SVM and AttnNet with various embeddings as input: Experiments with GloVe as baseline (**Bold+Underline**: highest; **Bold**: next highest); (*JointAff2vec: Chawla et al. (2019) reports results only for SST2 and SST5; **SWE method is not applicable for GloVe)