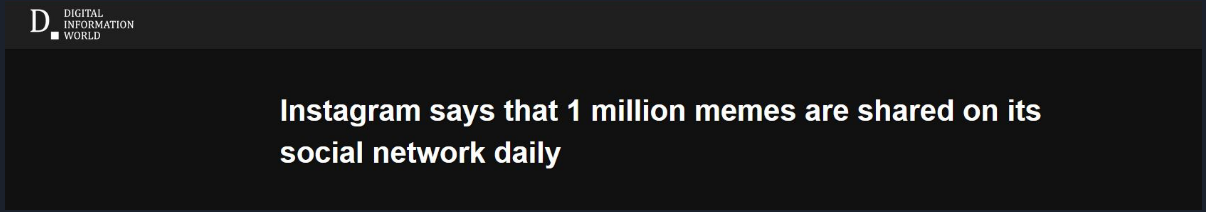
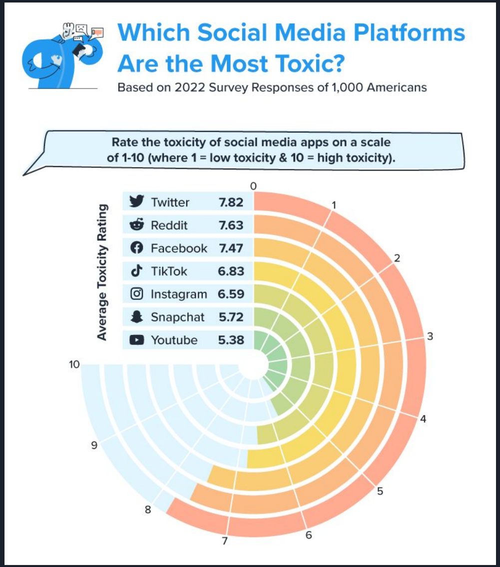
Classifying toxic memes with AI

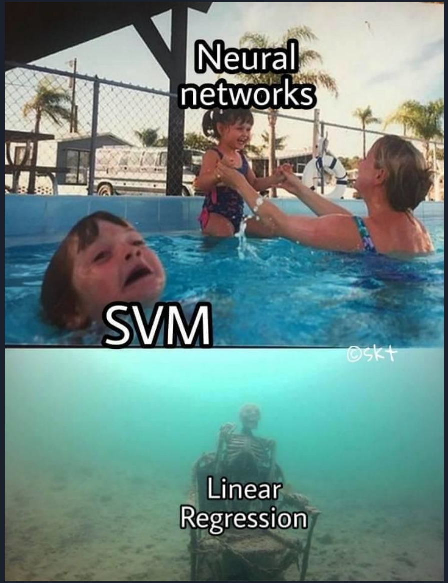
Group 4: Nesara Eranna Bethur, Janak Sharda, James Read, Nicholas Zhang

Modern communication through memes

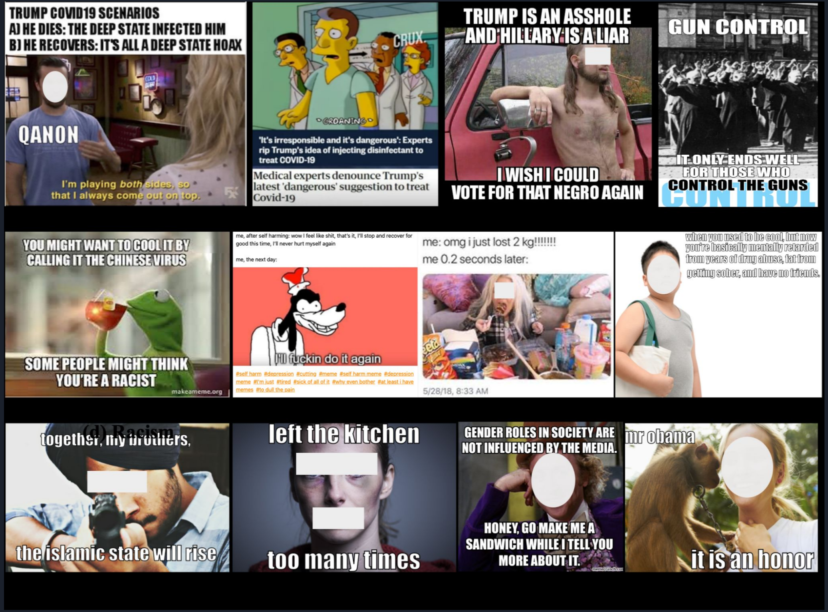

DIGITAL INFORMATION WORLD

Instagram says that 1 million memes are shared on its social network daily





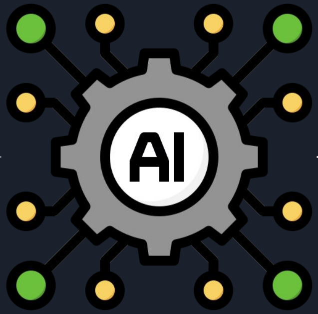
Toxic memes can influence the masses

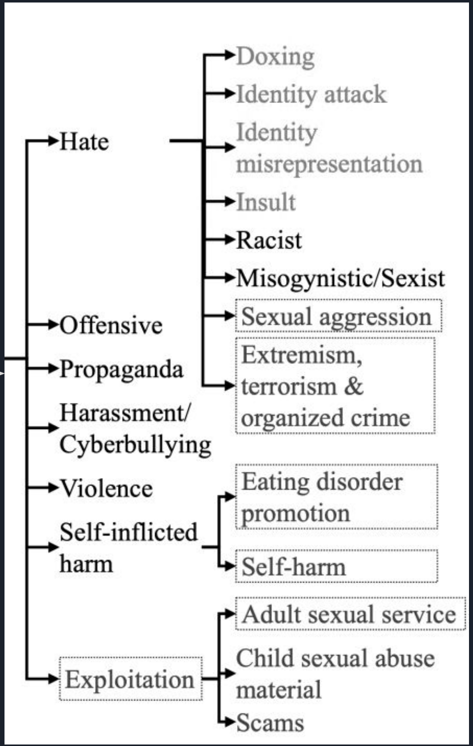


From [2]

From [4]

Classifying toxic memes can combat hate speech and improve mental health

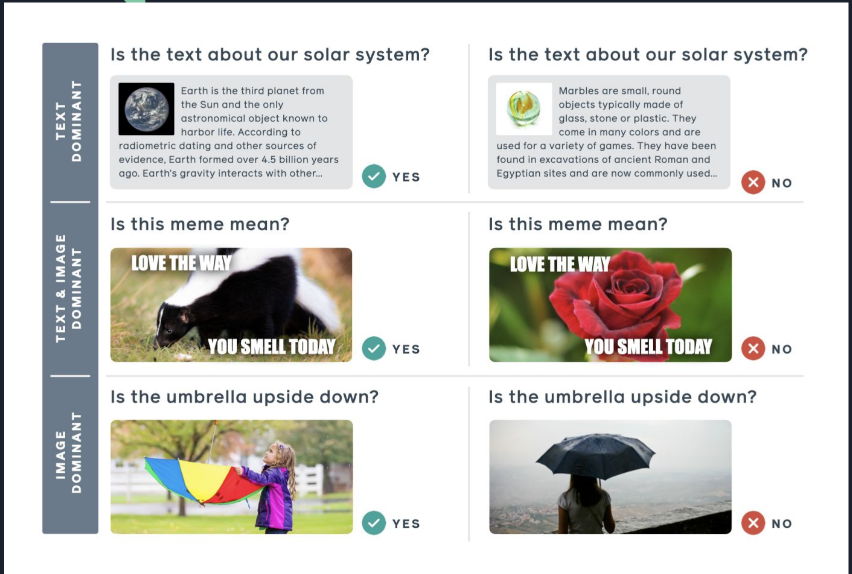




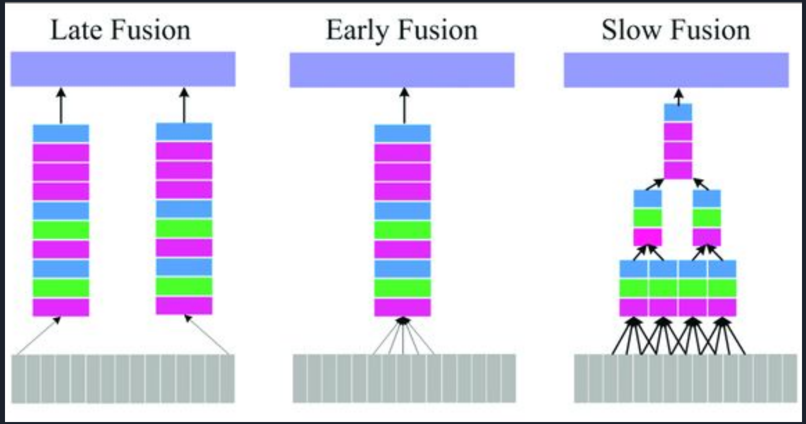
From [4]

[Vector Icons and Stickers - PNG, SVG, EPS, PSD and CSS (flaticon.com)](flaticon.com)

Input data is Multimodal



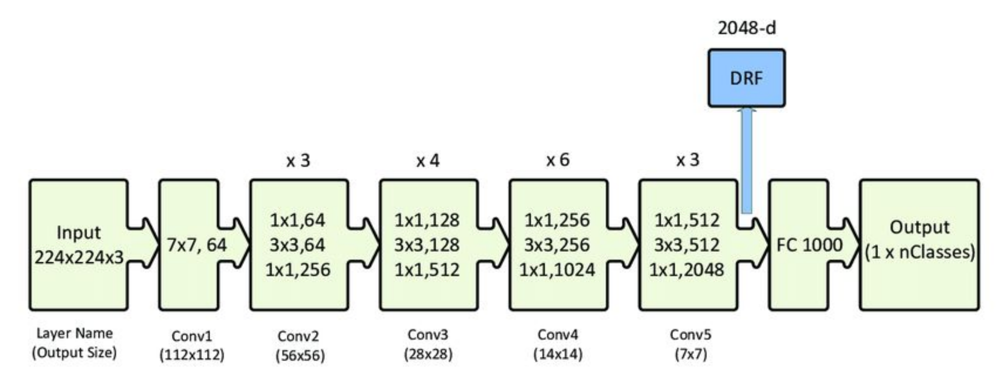
Both text and image are necessary to classify



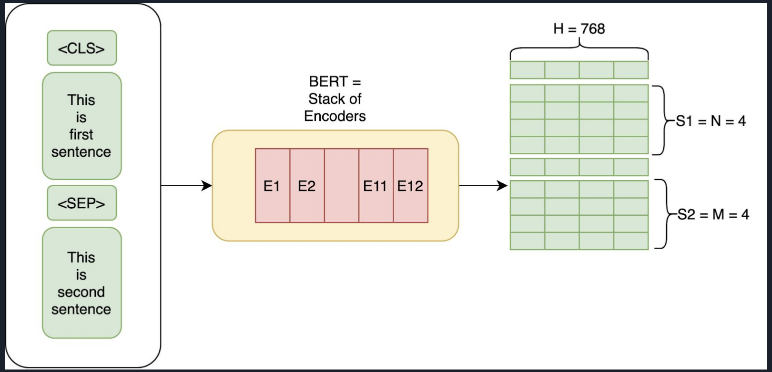
Various fusion techniques like early and late fusion can be used.

https://ai.facebook.com/tools/hatefulmemes/

Feature extraction through Pre-trained models



[5] ResNet



[6] BERT

Unsupervised Learning

BERT Layer	Homogeneity score
Layer-1	0.0056
Layer-7	0.0078
Layer-11	0.0169
Layer-12	0.0147

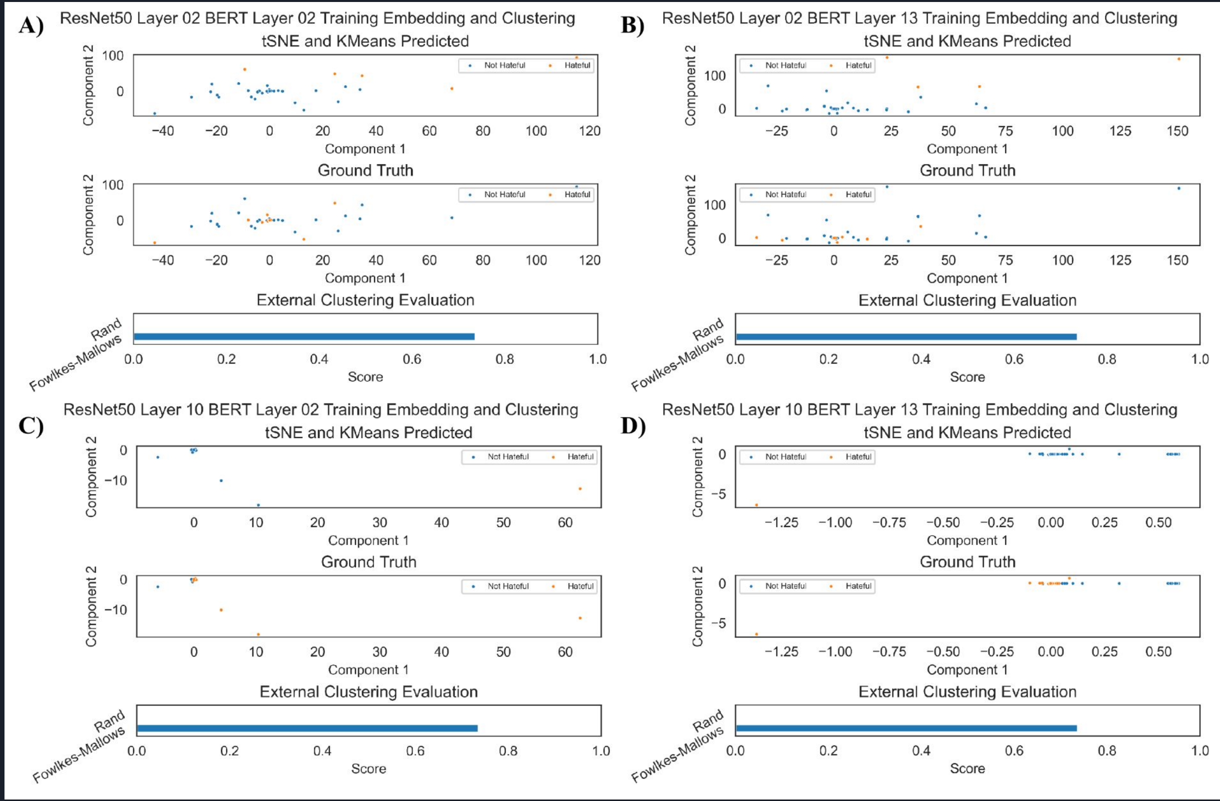
	Text only	Image Only	Fusion
FC,Layer11	0.0169	0.016	0.022

Unsupervised Learning

	Early(Layer-1)	Middle(Layer-7)	Late(Layer-11)
Early(Layer -2)	2.38e-5	2.1e-5	2.91e-6
Late(Layer-10, FC)	0.0070	0.0103	0.022

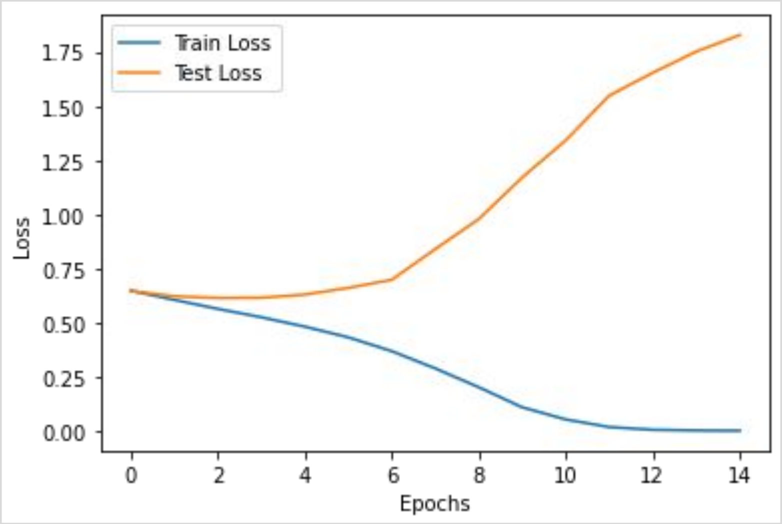
Layers	Homogeneity score
11	0.0169
11,12,13	0.0157
11,12	0.0167

Visualization



Supervised Learning

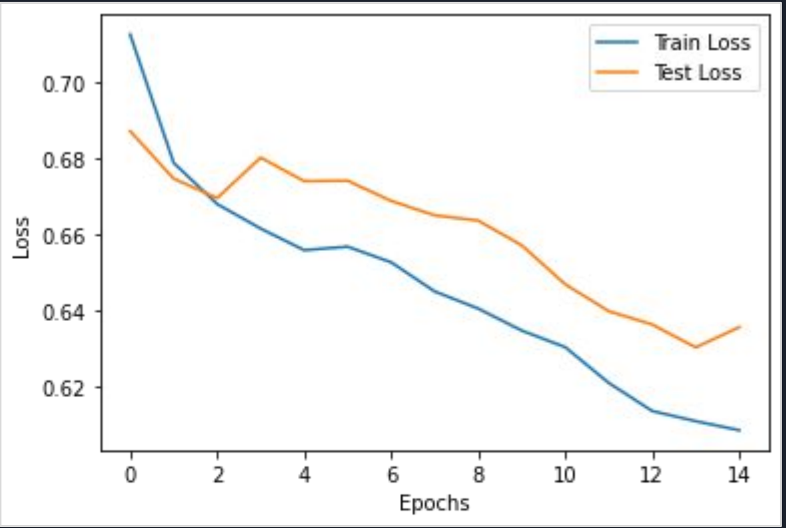
```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_40 (Dense)            (None, 256)               452608

 leaky_re_lu_32 (LeakyReLU)  (None, 256)               0

 dense_41 (Dense)            (None, 256)               65792

 leaky_re_lu_33 (LeakyReLU)  (None, 256)               0

 dense_42 (Dense)            (None, 256)               65792

 leaky_re_lu_34 (LeakyReLU)  (None, 256)               0

 dense_43 (Dense)            (None, 256)               65792

 leaky_re_lu_35 (LeakyReLU)  (None, 256)               0

 dense_44 (Dense)            (None, 256)               65792

 leaky_re_lu_36 (LeakyReLU)  (None, 256)               0

 dense_45 (Dense)            (None, 1)                 257

=================================================================
Total params: 716,033
Trainable params: 716,033
Non-trainable params: 0
_________________________________________________________________
```



Best AUROC Score: 0.706

Regularization

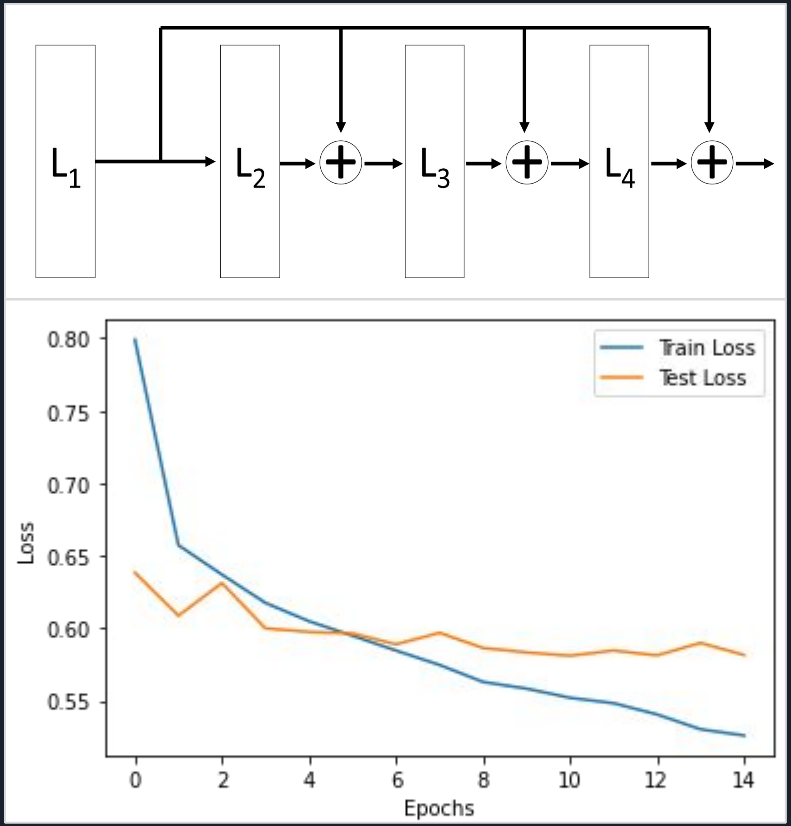
```
Model: "sequential_9"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_46 (Dense)            (None, 256)               452608

 leaky_re_lu_37 (LeakyReLU)  (None, 256)               0

 dropout_15 (Dropout)        (None, 256)               0

 dense_47 (Dense)            (None, 256)               65792

 leaky_re_lu_38 (LeakyReLU)  (None, 256)               0

 dropout_16 (Dropout)        (None, 256)               0

 dense_48 (Dense)            (None, 256)               65792

 leaky_re_lu_39 (LeakyReLU)  (None, 256)               0

 dropout_17 (Dropout)        (None, 256)               0

 dense_49 (Dense)            (None, 256)               65792

 leaky_re_lu_40 (LeakyReLU)  (None, 256)               0

 dropout_18 (Dropout)        (None, 256)               0

 dense_50 (Dense)            (None, 256)               65792

 leaky_re_lu_41 (LeakyReLU)  (None, 256)               0

 dense_51 (Dense)            (None, 1)                 257

=================================================================
Total params: 716,033
Trainable params: 716,033
Non-trainable params: 0
_________________________________________________________________
```



Best AUROC Score: 0.7174

Skip connections

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_13 (InputLayer)          [(None, 1767)]       0           []

 dense_64 (Dense)               (None, 256)          452608      ['input_13[0][0]']

 leaky_re_lu_52 (LeakyReLU)     (None, 256)          0           ['dense_64[0][0]']

 dropout_27 (Dropout)           (None, 256)          0           ['leaky_re_lu_52[0][0]']

 dense_65 (Dense)               (None, 256)          65792       ['dropout_27[0][0]']

 leaky_re_lu_53 (LeakyReLU)     (None, 256)          0           ['dense_65[0][0]']

 dropout_28 (Dropout)           (None, 256)          0           ['leaky_re_lu_53[0][0]']

 add (Add)                      (None, 256)          0           ['dropout_27[0][0]',
                                                                  'dropout_28[0][0]']

 dense_66 (Dense)               (None, 256)          65792       ['add[0][0]']

 leaky_re_lu_54 (LeakyReLU)     (None, 256)          0           ['dense_66[0][0]']

 dropout_29 (Dropout)           (None, 256)          0           ['leaky_re_lu_54[0][0]']

 add_1 (Add)                    (None, 256)          0           ['dropout_27[0][0]',
                                                                  'dropout_29[0][0]']

 dense_67 (Dense)               (None, 256)          65792       ['add_1[0][0]']

 leaky_re_lu_55 (LeakyReLU)     (None, 256)          0           ['dense_67[0][0]']

 dropout_30 (Dropout)           (None, 256)          0           ['leaky_re_lu_55[0][0]']

 add_2 (Add)                    (None, 256)          0           ['dropout_27[0][0]',
                                                                  'dropout_30[0][0]']

 dense_68 (Dense)               (None, 256)          65792       ['add_2[0][0]']

 leaky_re_lu_56 (LeakyReLU)     (None, 256)          0           ['dense_68[0][0]']

 dropout_31 (Dropout)           (None, 256)          0           ['leaky_re_lu_56[0][0]']

 add_3 (Add)                    (None, 256)          0           ['dropout_27[0][0]',
                                                                  'dropout_31[0][0]']

 dense_70 (Dense)               (None, 256)          65792       ['add_3[0][0]']

 leaky_re_lu_58 (LeakyReLU)     (None, 256)          0           ['dense_70[0][0]']

 dense_71 (Dense)               (None, 1)            257         ['leaky_re_lu_58[0][0]']

==================================================================================================
Total params: 781,825
Trainable params: 781,825
Non-trainable params: 0
__________________________________________________________________________________________________
```



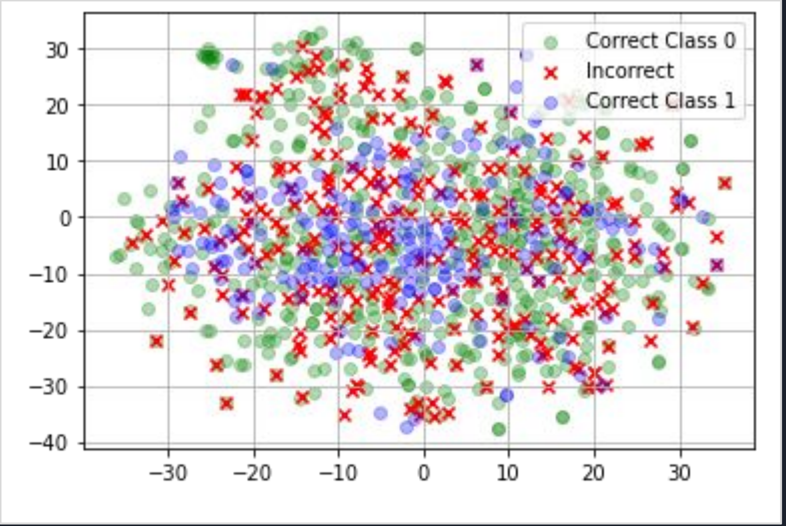
Best AUROC Score: 0.7460

Fusion results

Metric\Model	Late-Early	Late-Middle	Late-Late
Max AUROC score	0.7172	0.7172	0.7460
Precision	0.6068	0.5882	0.6319
Recall	0.5504	0.5943	0.5633

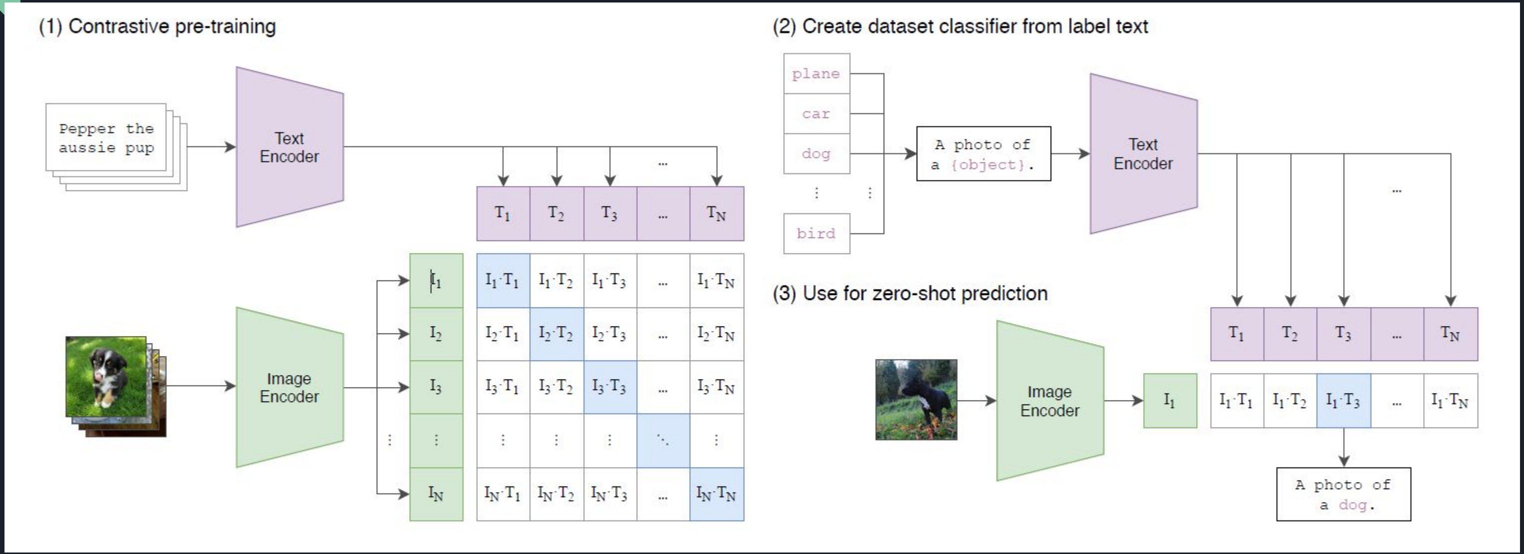
Bagging

1. 10 models, 3 layers text features + last layer image features.
2. Improvement in total score.
3. Improvement quantifies the relevant new information added by other layers
4. Best AUROC Score: 0.7624



tSNE plot of features

CLIP

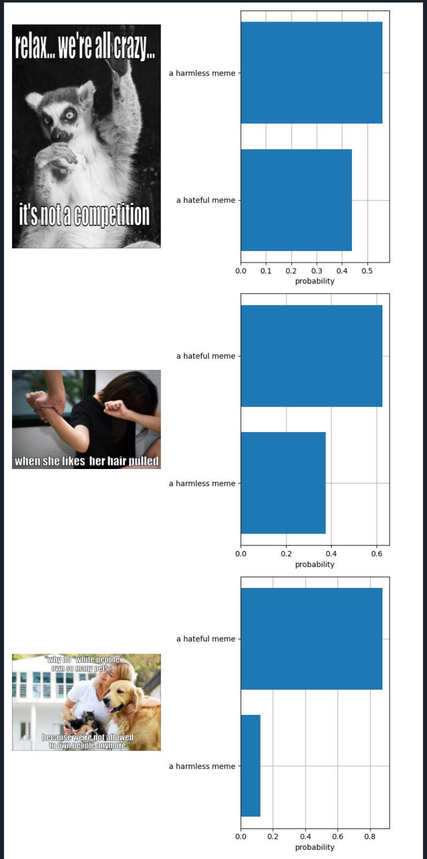


[7] Summary of CLIP approach

CLIP

Text-Only Accuracy (%)	Text-Only AUROC
50.3	0.49

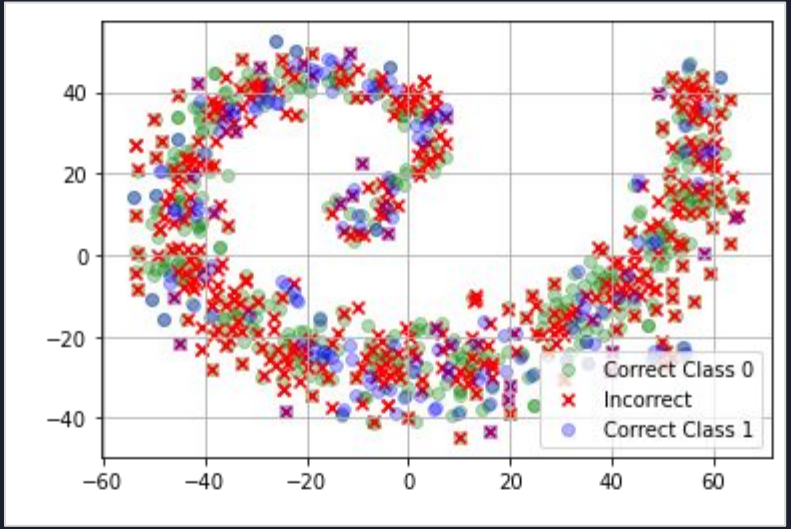
Internal Image Encoder	Image-Only Accuracy	Image-Only AUROC
Modified ResNet-50	51.1	0.29
Custom Vision Transformer	51.6	0.20



CLIP classification on Hateful Memes dataset

Bagging results on CLIP

1. 10 models, 3 layers text features + last layer image features from both networks.
2. Decrease in total score.
3. Degradation quantifies the new unimportant information added by other layers
4. Best AUROC Score: 0.7638 with bagging vs 0.7744 with just one good combination.



tSNE plot of features

Conclusion

- Using one set of features is insufficient for hateful meme classification
- Fusing encoded text and image features can solve this problem
- Training a dense neural network with late-late fusion provides best results
- Employing bagging like technique for choosing feature embeddings can provide relevance of features from different layers.
- With further fine-tuning, such a network could assist moderators to filter out hateful content on social media websites

Leaderboard

Hateful Memes: Phase 2

HOSTED BY FACEBOOK

Muennighoff	2	0.8310	0.6950	2020-10-31 23:34:40	1
HateDetectron	3	0.8108	0.7650	2020-10-16 23:02:31	1
kingsterdam	4	0.8053	0.7385	2020-10-31 23:20:27	3
burebista	5	0.7943	0.7430	2020-10-30 09:38:08	3
naoki	6	0.7886	0.7305	2020-10-31 04:43:28	3
MemeLords	7	0.7884	0.7450	2020-10-31 23:39:13	3
AiTingting	8	0.7848	0.7295	2020-10-31 12:56:43	3
mobot	9	0.7832	0.7320	2020-10-28 02:46:48	3
james005	10	0.7814	0.7280	2020-10-31 20:28:47	3
hate-alert	11	0.7808	0.7270	2020-10-26 13:13:22	3
mrsio	12	0.7806	0.7430	2020-10-20 16:30:18	3
letsgo	13	0.7801	0.7285	2020-10-28 12:51:03	3
QMUL-NUAA	14	0.7784	0.7300	2020-10-28 05:46:55	3
xyxyxxxy	15	0.7780	0.7270	2020-10-28 05:17:36	3
slawekbiel	16	0.7767	0.7320	2020-10-31 20:21:56	3
curvefitters	17	0.7731	0.7285	2020-10-31 00:59:48	2
nickyi	18	0.7654	0.7195	2020-10-31 22:50:22	3

References

[1] Pramanick, S., et al. "MOMENTA: A Multimodal Framework for Detecting Harmful Memes and Their Targets." arXiv preprint arXiv:2109.05184 (2021).

[2] Dimitrov, D., et al. "Detecting propaganda techniques in memes." arXiv preprint arXiv:2109.08013 (2021).

[3] Kiela, D., et al. "The hateful memes challenge: Detecting hate speech in multimodal memes." Advances in Neural Information Processing Systems 33 (2020): 2611-2624.

[4] Sharma, S., et al. "Detecting and Understanding Harmful Memes: A Survey." arXiv preprint arXiv:2205.04274 (2022).

[5]He, K., et. al. ”Deep Residual Learning for Image Recognition”, arXiv preprint arXiv: 1512:03385

[6] Devlin, J., et. al. ”BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, arXiv preprint arXiv: 1810:04805

[7] Redford, A., et. al. “Learning Transferable Visual Models From Natural Language Supervision” arXiv preprint, arXiv:2103.00020

[8] Mogadala, A. et al. "Trends in integration of vision and language research: A survey of tasks, datasets, and methods." Journal of Artificial Intelligence Research 71 (2021): 1183-1317.

[9] Radford et al. “Learning Transferable Visual Models From Natural Language Supervision.” International Conference on Machine Learning (2021).