Fast and precise PepPI predictions with ColabFold 1.5 and AlphaFold-Multimer v3

Yoann Battu
No affiliation
yoann.battumail@gmail.com

Abstract

ColabFold 1.5, based on AlphaFold-Multimer v3, allows for fast and precise transfer to prediction of Peptide-Protein Interactions, with 98% average Precision for 48% average Recall with only 2 seeds. Many features are tested, based both on confidence metrics and the distogram, as well as simple ML models. A transfer confidence metric is studied to support the transfer task. Features are also created on the MSA embeddings to test their use for data clustering, notably finding that a sequence-analog peptide-protein split still persists at the end of the AlphaFold pipeline.

1 Introduction, Related work

DeepMind significantly impacted computational biology when submitting their protein folding deep-learning model AlphaFold 2[1] (hereafter referred to as AF2) to the public, community-driven benchmarking competition CASP 14[2], achieving first place with a significant performance, both in relation to other contestants and to winners of the previous contests.

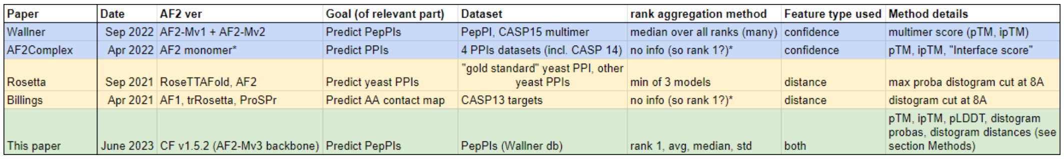
Much work has since been done to evaluate AF2 on many different direct or transfer tasks, and this work focuses on the use of AF2 for the downstream application of predicting whether two protein chains interact with each other, more specifically a pair of one peptide and one protein.

It was found soon after the release of the first AF2 version, that had only been trained on monomers, that the model could be used to predict the structures of multimers, first by fitting the two sequences into the single sequence input with either a glycine linker[3] or a gap, then by manipulating the *residue index*[4].

DeepMind then re-trained a new version of AF2 that took into account multimers, AF2-Multimer[5], which performed better than the earlier solutions and non-AF2 alternatives at predicting the structure of complexes.

The community and DeepMind further improved the AF2-Multimer models and their overall orchestration, leading to the release of the v2, and v3 in late December 2022[6]. These new model weights and changes to the higher-level options for working with the data and the models were integrated by the ColabFold team[7] over time, who maintain a public repository that allows running AF2 (all main versions) and some of its alternatives (ESMFold, OmegaFold, RoseTTAFold2) in a Google Colab environment. These are not exact reproductions as parts of the system are changed, most importantly the sequence databases and search algorithm used for the MSA generation[8].

The main work on using AF2 for the sake of PepPI prediction inspiring this one is Wallner's work[9] published in September 2022, in which he studies the effect of sampling a massive number of repetitions (up to thousands for his CASP15 contest entry) on the quality of the generated structure of complexes (as well as the reliability of its confidence metrics), and also evaluated how well one of AF2's confidence metrics could be used to predict a binary interaction label. Wallner ranked third in



Paper	Date	AF2 ver	Goal (of relevant part)	Dataset	rank aggregation method	Feature type used	Method details
Wallner	Sep 2022	AF2-Mv1 + AF2-Mv2	Predict PepPIs	PepPI, CASP15 multimer	median over all ranks (many)	confidence	multimer score (pTM, ipTM)
AF2Complex	Apr 2022	AF2 monomer*	Predict PPIs	4 PPIs datasets (incl. CASP 14)	no info (so rank 1?)*	confidence	pTM, ipTM, "Interface score"
Rosetta	Sep 2021	RoseTTAFold, AF2	Predict yeast PPIs	"gold standard" yeast PPI, other yeast PPIs	min of 3 models	distance	max proba distogram cut at 8A
Billings	Apr 2021	AF1, trRosetta, ProSPr	Predict AA contact map	CASP13 targets	no info (so rank 1?)*	distance	distogram cut at 8A
This paper	June 2023	CF v1.5.2 (AF2-Mv3 backbone)	Predict PepPIs	PepPIs (Wallner db)	rank 1, avg, median, std	both	pTM, ipTM, pLDDT, distogram probas, distogram distances (see section Methods)

Figure 1: Comparison of the 4 main references and this work.

CASP15 in the multimer category with this method, as per the results released in December 2022. A paper and video are available on the use of this method specifically for the CASP 15 data[10][11].

Some other works that use AF2 or a similar model to transfer to either PepPI or PPI have their own methods, such as AF2Complex[12] and a paper by the Rosetta team[13]. Billings et al.[14] mention a way to obtain a contact probability map.

These 4 sources vary in many ways, these differences are summarized in figure 1.

These differences naturally raise the question of which output of an AF2-like model to use, between the various confidence metrics and distance measures available, and how they compare for transfer to PepPIs when evaluated against the same dataset.

Beyond the confidence metrics and the 3D distogram, one type of output that AF2 provides that has not been found to be much used in the literature yet is the neural network embeddings that AF2 has learned to generate in its structure-generation process. Throughout its pipeline, AF2 updates and jointly embeds a representation of the MSA and a pairwise representation of the amino acids.

Before the last call to the structure generation module and the output of the last 3D structure, these two neural embeddings contain (at least) all the information necessary to generate the output structure. These structure-aware tensor embeddings can thus potentially serve for any task that NN embeddings are typically used for in recent years (input to a downstream task, data viz, clustering, interfacing models) or for a biology or chemistry tasks that previously used Fingerprints or embeddings from AEs.

As such, this paper aims to :

- formulate a feature extraction schema to compare the various existing transfer methods,
- build new features based on the intuitions of the reference papers,
- evaluate all options (including reproductions or similar pipelines to the reference papers) against one dataset,
- evaluate using basic ML models to transfer from a few of the features,
- create a simple transfer confidence measure and evaluate its correlation to performance,
- build new features on the MSA embeddings to try using them for data visualization, nuancing the performance evaluation or serving as a confidence metric,
- evaluate the performance of the recent AF-Multimer v3 and ColabFold v1.5.2 on this task, especially with a skew towards fast memory-light computation costs.

2 Methods, Materials

2.1 Dataset

The dataset chosen for this work is the one made available by Wallner, with some minor changes. This dataset originates, as I understand, from the PepBDB[15] database which Lei[16] reduced to another dataset in their work. Wallner then further reduced that version through clustering by only keeping one protein or peptide per ECOD family[17].

As only the PDB ids and chain ids are given, some PDB entries have more than one possible solution for which chain could be referred to, even when considering that one is a peptide. Such cases were left out for safety, for a final working dataset of 106 positive interaction complexes and 526 negative complexes, generated by Wallner by taking non-existing pairs out of the possible combinations. The corresponding sequences were taken from current day PDB, including missing residues, and not PepBDB.

2.2 ColabFold run + Feature extraction

The complex sequences are fed into ColabFold v1.5.2 (parameters available in Supplementary file). The system makes 1 to 10 predictions per input (2 RNG seeds x 5 models, for 10 max predictions made, unless stopped by the early stopping), then re-ranks them according to the multimer score. These multiple predictions per single input will hereafter be referred to as "ranks" to not confuse them with "model" or "predictions" when it could be unclear.

For every rank, is saved :

- the final structure,
- 7 2D maps obtained from the 3D distogram,
- the MSA single representation,
- the confidence metrics.

These parameters amount to an output zip of 3 to 150 MB per complex, computed in about 10 to 20 mins when running all 10 ranks (on Colab Pro +, i.e an A100 GPU) including the MMseqs server total response time.

The referenced papers vary in what information they extract from running a form of AF2 or of an AF2-like model, to use as a variable to predict interaction from. Taking their intuitions, comparing their differences and analyzing all the possible outputs allow for the identification of many possible features to evaluate.

Overall, 161 features are built (details in Supplementary) with the key points being :

- The pTM, ipTM, plDDT, the distance map, and the contact probabilities are used.
- For every feature per rank, they are aggregated to a single value per complex by either taking the first rank, the average, the median or the standard deviation over all ranks.
- The plDDT is split into a `pep_plddt`.
- Wallner's method is reproduced as one of them (`median_ranks_multimer`).
- From the 3D distogram, multiple contact probability maps are extracted, corresponding to the classic 8Å threshold (slice 18) but also neighboring bins, overall ranging approximately from 7.3Å to 8.9Å (slice 16 to 21).
- Only the inter-object sections of the distogram are used, and not the intra-object ones.

The operations of the computed features on the confidence metrics and on the 3D distogram are summarized in figure 2.

2.3 F-Beta based threshold calibration

When encountering a set of prediction probabilities, the decision threshold to apply for classification is automatically calibrated to maximize a specific F-Beta score, which was somewhat arbitrarily set to 5:1 for Precision:Recall, although a 10:1 ratio was also tested without much difference in the results of the rankings of the methods.

2.4 Basic ML models

Evaluating the discrimination power of each feature by itself naturally raises the question of ensembling, and of the performance of an ML model taking as input multiple of these features.

Considering the high performance of some features by themselves (see Results), the main motivation behind testing some ML models here was to see if they would have less variance over runs by taking in features from different origins (iptm, plddt, distogram) rather than a performance increase. As such, the models tested were left at their near-default parameters, and only took in a manual sub-selection of features with their own high discrimination power and different origins.

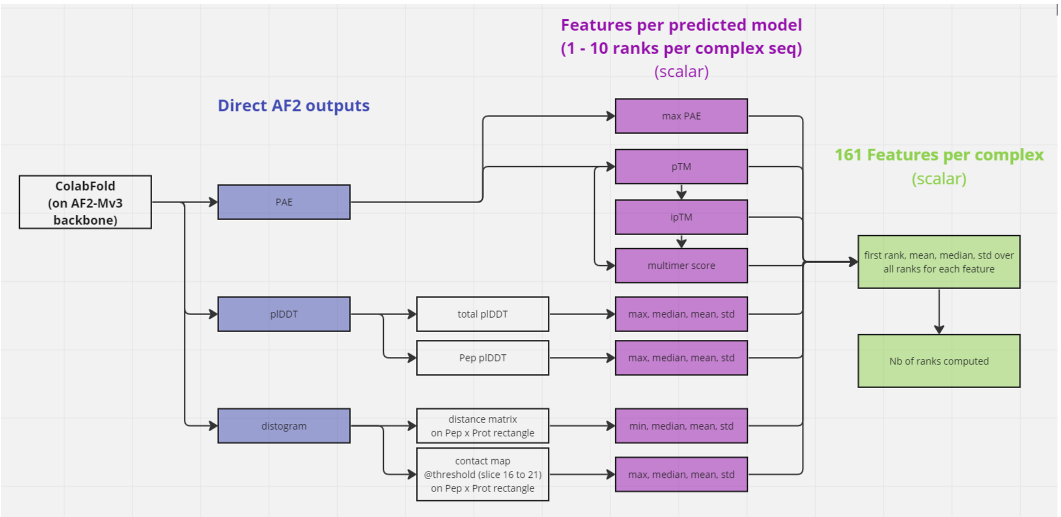

Features per predicted model
(1 - 10 ranks per complex seq)
(scalar)
Direct AF2 outputs
ColabFold (on AF2-Mv3 backbone)
PAE
plDDT
distogram
max PAE
pTM
ipTM
multimer score
total plDDT
Pep plDDT
distance matrix on Pep x Prot rectangle
contact map @threshold (slice 16 to 21) on Pep x Prot rectangle
max, median, mean, std
max, median, mean, std
min, median, mean, std
max, median, mean, std
161 Features per complex
(scalar)
first rank, mean, median, std over all ranks for each feature
Nb of ranks computed

Figure 2: Diagram of the computation process of the tested features.

It is also hoped that a model based on multiple features might generalize better to other datasets, although this is not evaluated here.

The ML models tested are a selection from the classifiers available in Scikit-Learn : Logistic Regression, K-Nearest Neighbours, Linear SVM, RBF SVM, Gaussian Process, Decision Tree, Random Forest, Multi-layer Perceptron, AdaBoost, Naive Bayes, Quadratic Discriminant Analysis.

2.5 Evaluation loop

For evaluating a method the dataset was split into a training and testing set by a stratified 5-fold cross-validation (no validation as there is no hyperparameter optimisation). Such a split is further repeated 622 times to try and trigger as many disfavorable train/test splits as possible.

For the single feature evaluation, inside each of the splits the F-Beta threshold calibration is applied on the train set, and evaluated on the test set. For the evaluation of the ML models inside each split the training set is standardized, the model fitted on it, then the decision threshold calibrated in the same way as each feature was (on the same train set), and then the overall pipeline of scaling - model - threshold calibration is evaluated on the test set.

Over all these loops, the performance is aggregated for comparison of the options, through the average, standard deviation, median, and percentiles to account for the center 50%, 75%, 90% and 95% of the values.

2.6 Confidence measure of transfer

Inspired by AF2's own self-estimates of accuracy (plDDT, pTM), and Wallner's effort of studying their correlation to actual performance for peptide - protein complexes, it seems valuable to add a measure of confidence for this work's transfer method. As the confidence measures of the structure prediction task are taken for the transfer to the downstream task of predicting the PepPIs, they are not counted as viable candidates for a transfer confidence measure.

Instead, the naive approach of using the distance between a point and the auto-calibrated decision threshold is tested out. The distances, and whether or not each point is successfully classified, are tracked for every test set computed in the evaluation loop, and then summarized by binning the distances and aggregating the distribution of successes and failures.

2.7 MSA embeddings

Out of the total final MSA embedding that exists in the model, only the first slice is returned (and used for recycles), of shape `(n_AA, n_channels)`, with 256 NN channels. From this 2D data, a few options are used to summarize things down to 1D, to remove the variation of sequence length over complexes.

One could either summarize the entire row, just the part that used to correspond to the peptide in the input or the part of the protein. In this work, all three options are combined against every summarization option that follows. There is no guarantee that, after running the entire model, a vertical separation remains in the information but as the shape returned is the shape of the sequence, it seems interesting to try to check.

For each such section of a row, the values are summarized by taking either the mean, the minimum, the maximum or the standard deviation. Each combination of a subsection selection with a summarization method is run for the 1-10 ranks and then further aggregated over them by one of the same options as the confidence or distance features : taking the first rank, the median, the mean or the standard deviation over ranks.

3 Results

3.1 Single feature classification

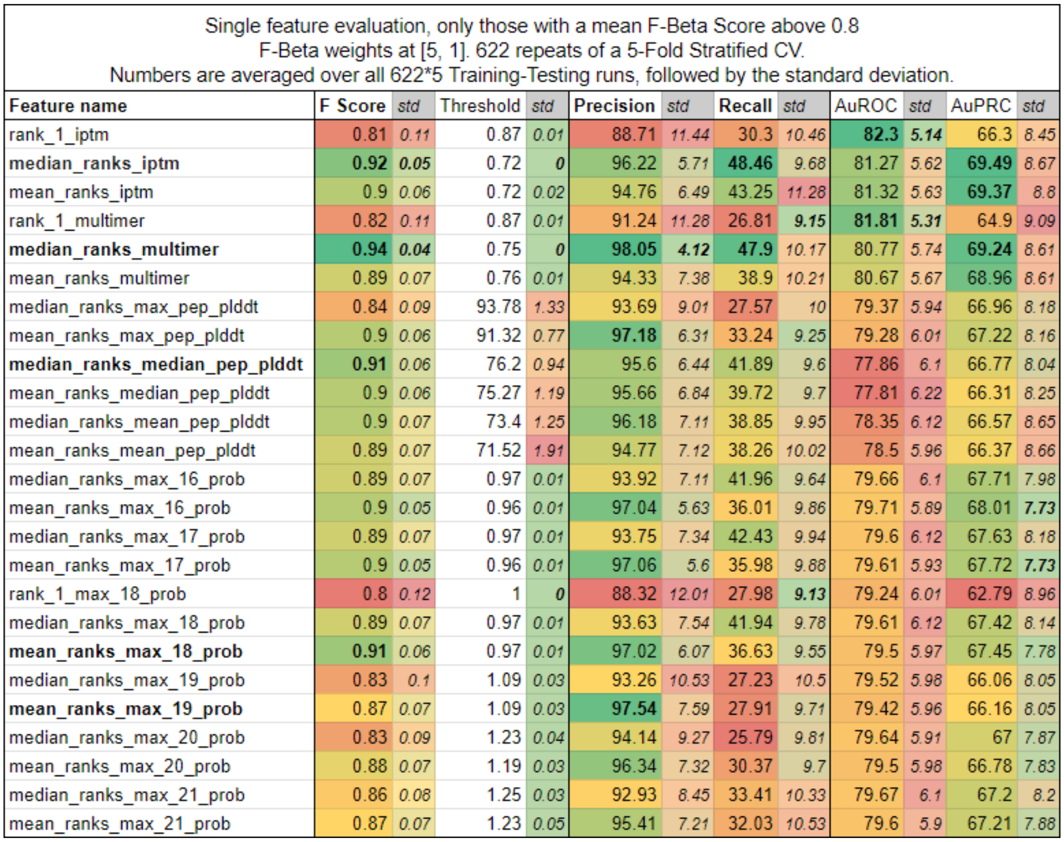
For each single feature, the metrics used to track performance are the F-Beta score (weighted 5 to 1 towards Precision), the Precision and the Recall of the decision threshold that maximizes the F-Beta. Also indicated is the threshold itself, to check for variation.

It is typical to show ROC Curves, Precision-Recall Curves, and their corresponding Area under the curve, but here it is more interesting to evaluate the ability to solve the task evaluated for the specified objective function optimally than to compare the overall decision power of each feature for any objective function. Nonetheless, the two curves will be available for every feature as supplementary information, and the areas under the curves are also indicated in the scores.

From the results in figure 3, some observations can be made as to the performance of the features extracted from the confidence scores and the distances :

- To aggregate over ranks, the std is never good and does not even appear, and taking the rank 1 is always worse than taking either the median or the mean over all ranks. For these high performance features, the mean is always better than the median for the `max_pep_plddt` and the distance slices, and inversely for the ipTM, multimer and rest of the `pep_plddt` features.
- Both the confidence-based and the distance-based sections have high performance features, although the best ones are confidence-based.
- The feature with the highest F-Beta, highest Precision, lowest Precision variance, and lowest threshold variance is the same one, the `median_ranks_multimer` (a.k.a the reproduction of Wallner's method). Its Recall is near the highest too, for an average test performance of about 98% Precision and 48% Recall over 622*5 runs.
- The multimer score is better than both its components, the ipTM and pTM, although the ipTM is close behind while the pTM features do not even appear. A different weighting than the existing 80/20 may be more optimal.
- The various slices of the distogram are close to one another in performance, although their recall is often considerably worse than the pTM and multimer options. Doing the same comparison on slices much further apart in Angstrom from one another may have more information.
- The 4 top performance features of `[median/mean]_ranks_[iptm/multimer]` have the highest average AuPRC scores, although not necessarily in the same orders, and large variations of performance are not necessarily reflected in the AuPRC score, as is the case for median of multimer and mean of multimer.
- The AuROC scores overall are not quite aligned with the performance, and the highest one is a notably worse option when comparing the optimal P-R compromise.

Additionally, analyzing the percentile values shows that in 50% of the 622x5 runs, `median_ranks_multimer` is at 100% Precision. In 95% of them, it is still above 85%.



Single feature evaluation, only those with a mean F-Beta Score above 0.8
F-Beta weights at [5, 1]. 622 repeats of a 5-Fold Stratified CV.
Numbers are averaged over all 622*5 Training-Testing runs, followed by the standard deviation.

Feature name	F Score	*std*	Threshold	*std*	Precision	*std*	Recall	*std*	AuROC	*std*	AuPRC	*std*
rank_1_iptm	0.81	*0.11*	0.87	*0.01*	88.71	*11.44*	30.3	*10.46*	**82.3**	***5.14***	66.3	*8.45*
median_ranks_iptm	**0.92**	***0.05***	0.72	***0***	96.22	*5.71*	**48.46**	*9.68*	81.27	*5.62*	**69.49**	*8.67*
mean_ranks_iptm	0.9	*0.06*	0.72	*0.02*	94.76	*6.49*	43.25	*11.28*	81.32	*5.63*	**69.37**	*8.8*
rank_1_multimer	0.82	*0.11*	0.87	*0.01*	91.24	*11.28*	26.81	***9.15***	**81.81**	***5.31***	64.9	*9.09*
median_ranks_multimer	**0.94**	***0.04***	0.75	***0***	**98.05**	***4.12***	**47.9**	*10.17*	80.77	*5.74*	**69.24**	*8.61*
mean_ranks_multimer	0.89	*0.07*	0.76	*0.01*	94.33	*7.38*	38.9	*10.21*	80.67	*5.67*	68.96	*8.61*
median_ranks_max_pep_plddt	0.84	*0.09*	93.78	*1.33*	93.69	*9.01*	27.57	*10*	79.37	*5.94*	66.96	*8.18*
mean_ranks_max_pep_plddt	0.9	*0.06*	91.32	*0.77*	**97.18**	*6.31*	33.24	*9.25*	79.28	*6.01*	67.22	*8.16*
median_ranks_median_pep_plddt	**0.91**	*0.06*	76.2	*0.94*	95.6	*6.44*	41.89	*9.6*	77.86	*6.1*	66.77	*8.04*
mean_ranks_median_pep_plddt	0.9	*0.06*	75.27	*1.19*	95.66	*6.84*	39.72	*9.7*	77.81	*6.22*	66.31	*8.25*
median_ranks_mean_pep_plddt	0.9	*0.07*	73.4	*1.25*	96.18	*7.11*	38.85	*9.95*	78.35	*6.12*	66.57	*8.65*
mean_ranks_mean_pep_plddt	0.89	*0.07*	71.52	*1.91*	94.77	*7.12*	38.26	*10.02*	78.5	*5.96*	66.37	*8.66*
median_ranks_max_16_prob	0.89	*0.07*	0.97	*0.01*	93.92	*7.11*	41.96	*9.64*	79.66	*6.1*	67.71	*7.98*
mean_ranks_max_16_prob	0.9	*0.05*	0.96	*0.01*	97.04	*5.63*	36.01	*9.86*	79.71	*5.89*	68.01	***7.73***
median_ranks_max_17_prob	0.89	*0.07*	0.97	*0.01*	93.75	*7.34*	42.43	*9.94*	79.6	*6.12*	67.63	*8.18*
mean_ranks_max_17_prob	0.9	*0.05*	0.96	*0.01*	97.06	*5.6*	35.98	*9.88*	79.61	*5.93*	67.72	***7.73***
rank_1_max_18_prob	0.8	*0.12*	1	***0***	88.32	*12.01*	27.98	***9.13***	79.24	*6.01*	62.79	*8.96*
median_ranks_max_18_prob	0.89	*0.07*	0.97	*0.01*	93.63	*7.54*	41.94	*9.78*	79.61	*6.12*	67.42	*8.14*
mean_ranks_max_18_prob	**0.91**	*0.06*	0.97	*0.01*	97.02	*6.07*	36.63	*9.55*	79.5	*5.97*	67.45	*7.78*
median_ranks_max_19_prob	0.83	*0.1*	1.09	*0.03*	93.26	*10.53*	27.23	*10.5*	79.52	*5.98*	66.06	*8.05*
mean_ranks_max_19_prob	0.87	*0.07*	1.09	*0.03*	**97.54**	*7.59*	27.91	*9.71*	79.42	*5.96*	66.16	*8.05*
median_ranks_max_20_prob	0.83	*0.09*	1.23	*0.04*	94.14	*9.27*	25.79	*9.81*	79.64	*5.91*	67	*7.87*
mean_ranks_max_20_prob	0.88	*0.07*	1.19	*0.03*	96.34	*7.32*	30.37	*9.7*	79.5	*5.98*	66.78	*7.83*
median_ranks_max_21_prob	0.86	*0.08*	1.25	*0.03*	92.93	*8.45*	33.41	*10.33*	79.67	*6.1*	67.2	*8.2*
mean_ranks_max_21_prob	0.87	*0.07*	1.23	*0.05*	95.41	*7.21*	32.03	*10.53*	79.6	*5.9*	67.21	*7.88*

Figure 3: Results of the evaluation of the single features : F-Score, Real-value threshold, Precision, Recall, AuROC, AuPRC.

3.2 Basic ML classifiers

While in figure 4 the Recalls and AuPRCs are in general higher, the compromises found are not better Precision-Recall trade offs than the best single feature. It may be, though, that some of these generalize better to other datasets, and as such should be considered as viable options in other works.

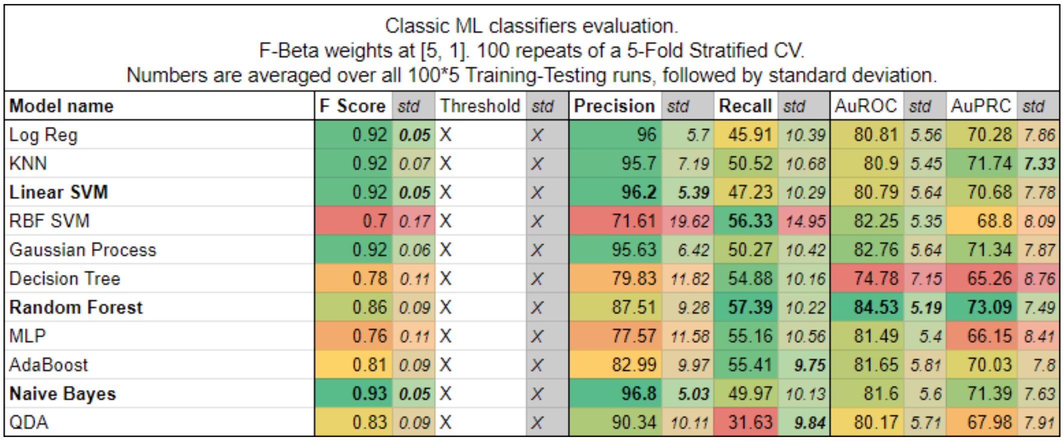
3.3 Confidence measure

For the purpose of safety and transparency, it seems relevant if not necessary to have a sort of confidence measure to the transfer itself to have a continuous value that accompanies predictions to indicate how “confident” the model is, or how similar to the training set our data might be.

The first naive approach of taking the distance to the threshold can be plotted after the performance evaluation loop, binning such distances together and computing the accuracy on each bin.

From figure 5, plotted over the `median_ranks_multimer` feature, the best single feature, it can be seen that the success is generally monotonous in increasing away from the threshold, apart from the very end, where the bottom 12.5% (that could be understood as the 12.5% unluckiest data splits) of the values reach down.

Nevertheless, this plot can serve as additional information on the reliability of a prediction, especially for the range right below the threshold where it might be safer to consider anything between 0 and -0.1 to be an indeterminate prediction, rather than a positive or negative interaction.



Classic ML classifiers evaluation.
F-Beta weights at [5, 1]. 100 repeats of a 5-Fold Stratified CV.
Numbers are averaged over all 100*5 Training-Testing runs, followed by standard deviation.

Model name	**F Score**	*std*	Threshold	*std*	**Precision**	*std*	**Recall**	*std*	AuROC	*std*	AuPRC	*std*
Log Reg	0.92	***0.05***	X	*X*	96	*5.7*	45.91	*10.39*	80.81	*5.56*	70.28	*7.86*
KNN	0.92	*0.07*	X	*X*	95.7	*7.19*	50.52	*10.68*	80.9	*5.45*	71.74	***7.33***
Linear SVM	0.92	***0.05***	X	*X*	**96.2**	***5.39***	47.23	*10.29*	80.79	*5.64*	70.68	*7.78*
RBF SVM	0.7	*0.17*	X	*X*	71.61	*19.62*	56.33	*14.95*	82.25	*5.35*	68.8	*8.09*
Gaussian Process	0.92	*0.06*	X	*X*	95.63	*6.42*	50.27	*10.42*	82.76	*5.64*	71.34	*7.87*
Decision Tree	0.78	*0.11*	X	*X*	79.83	*11.82*	54.88	*10.16*	74.78	*7.15*	65.26	*8.76*
Random Forest	0.86	*0.09*	X	*X*	87.51	*9.28*	**57.39**	*10.22*	**84.53**	***5.19***	**73.09**	*7.49*
MLP	0.76	*0.11*	X	*X*	77.57	*11.58*	55.16	*10.56*	81.49	*5.4*	66.15	*8.41*
AdaBoost	0.81	*0.09*	X	*X*	82.99	*9.97*	55.41	***9.75***	81.65	*5.81*	70.03	*7.8*
Naive Bayes	**0.93**	***0.05***	X	*X*	**96.8**	***5.03***	49.97	*10.13*	81.6	*5.6*	71.39	*7.63*
QDA	0.83	*0.09*	X	*X*	90.34	*10.11*	31.63	***9.84***	80.17	*5.71*	67.98	*7.91*

Figure 4: Results of the evaluation of the classic ML classifiers : F-Score, Precision, Recall, AuROC and AuPRC. The decision threshold is no longer included as it is no longer immediately applicable to the feature data, but just the model output.

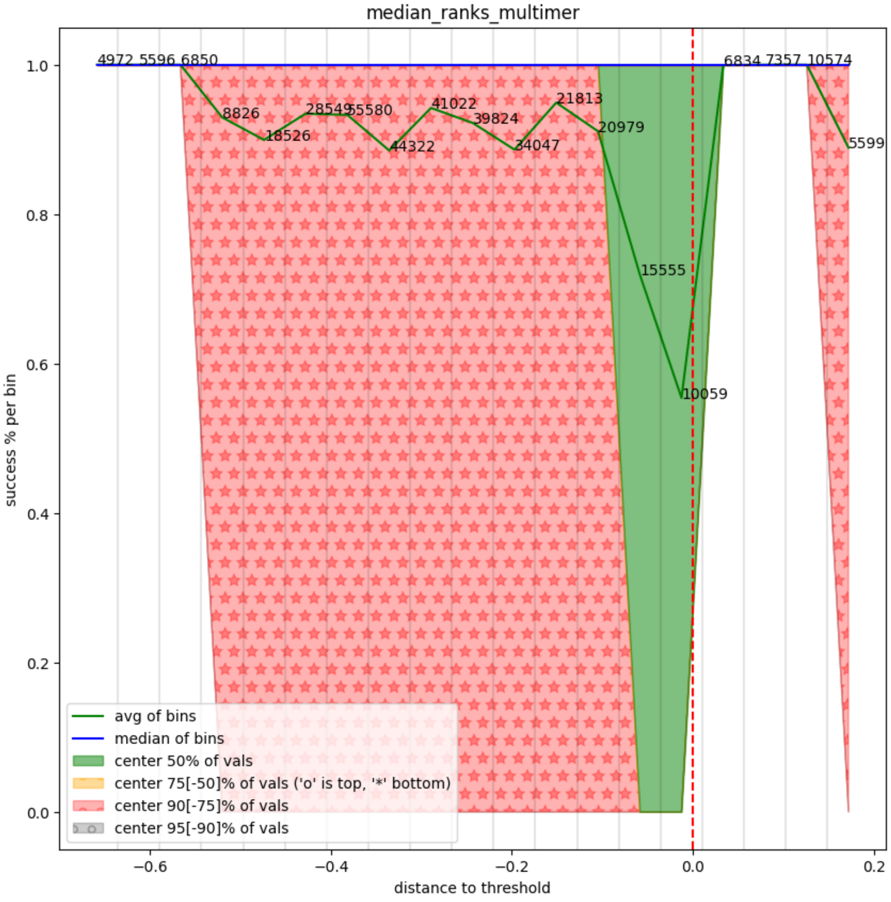


Figure 5: Variation of the success rate (accuracy metric) depending on the point's distance to the decision threshold. The number besides each plotted bin average is the sample size of the bin. The green and orange areas nearly do not appear as they are nearly always stuck to the top, at 100%.

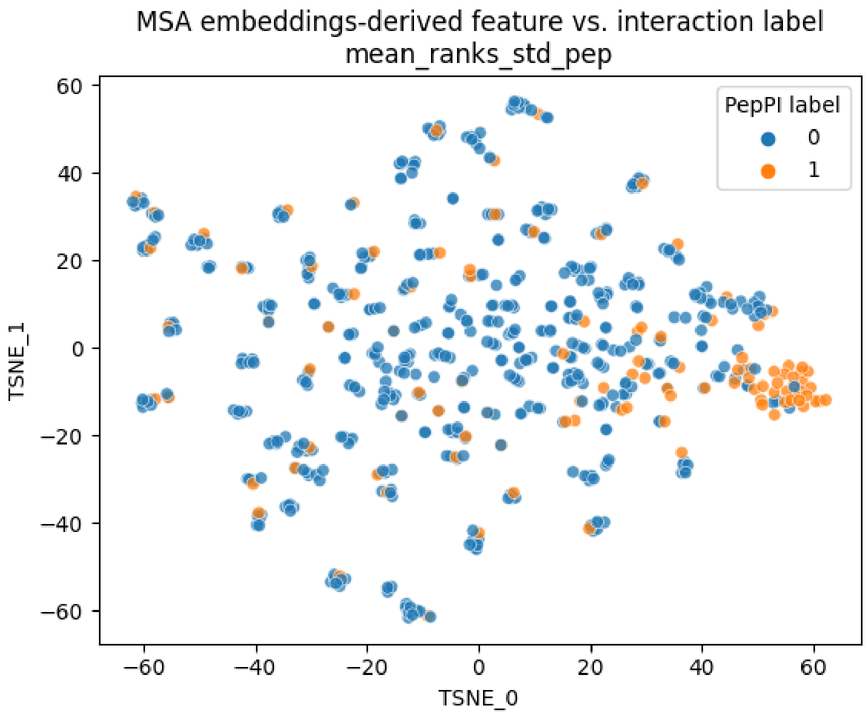


Figure 6: Standard deviation of the "peptide part" of the MSA embeddings, averaged over all ranks and fed into a tSNE visualization.

3.4 MSA embeddings

For the features derived from the MSA embeddings, each feature is a 256-length vector, the dataset was then fed to a tSNE visualization and the following observations made are just patterns observed between plots.

In regards to how well the MSA embeddings separate the interaction label, these observations can be made, and figure 6 is an example :

- Most features show nothing visually interesting, except 3 : `mean_pep`, `std_pep` and `min_pep`.
- Taking the std over all ranks is never useful while taking the median or the mean are often similar to each other, both slightly better at clustering than taking the rank 1.
- The useful features are all based on the peptide part of the MSA embeddings, and their counterparts based on the protein part or on the whole embedding give no result.
- `std_pep` may be the best feature, the first time taking the standard deviation as a feature is useful.

In regards to how well the MSA embeddings map to the multimer score feature, these observations can be made, and figure 7 is an example :

- Taking the std over all ranks is never useful, taking the mean, median or rank 1 give similar results to one another.
- `max_prot`, `max_pep`, `max_all_aa` and `std_prot` give nothing useful.
- `min_pep` and `std_pep` cluster well high multimer values, but not the low ones
- `mean_pep` cluster both low and high multimer values decently well
- The 5 other features cluster well low multimer values, but not the high ones

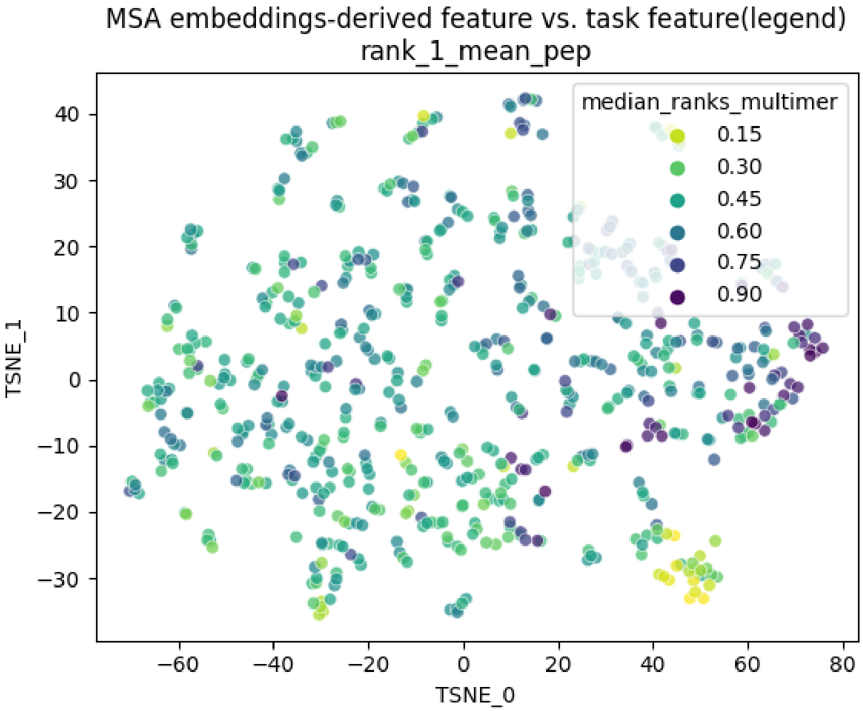


Figure 7: Average of the "peptide part" of the MSA embeddings, first rank only, fed into a tSNE visualization.

- Features that cluster well high multimer values are all based on the peptide part, those that cluster well low values are all based on either the protein counterpart or the entire embedding (except for `mean_pep`).

In regards to how well the MSA embeddings map to the transfer performance of the multimer score feature, most features add no new information visually, except for a few :

- `mean_pep`, `std_pep` and `min_pep` once again have interesting clusters, as they cluster True Positives fairly well.
- Taking the standard deviation over all ranks does nothing, taking the rank 1, median or mean over ranks are generally close, although here the order varies between features.
- Both `mean_pep` and `min_pep` have the single False Positive inside of the True Positive cluster, while `std_pep` puts it far away in the tSNE space.

Figure 8 is an example.

4 Structure of the 4 extreme cases

Since AF2's original task is to predict structures, it would be a shame to not use them at all, and so the rank 1 structures of the 4 most extreme cases have been selected to be displayed, along with their identifier These are figures 9 through 12. By extreme case is implied the best True Positive (highest multimer score complex with a positive label), worse False Positive (highest scoring negative label, predicted as positive), worse False Negative and best True Negative.

The complexes are either named with the format `PdbId_PepChainId-ProtChainId` for positive labels, or `PepPdbId_PepChainId-ProtPdbId_ProtChainId` for negative ones.

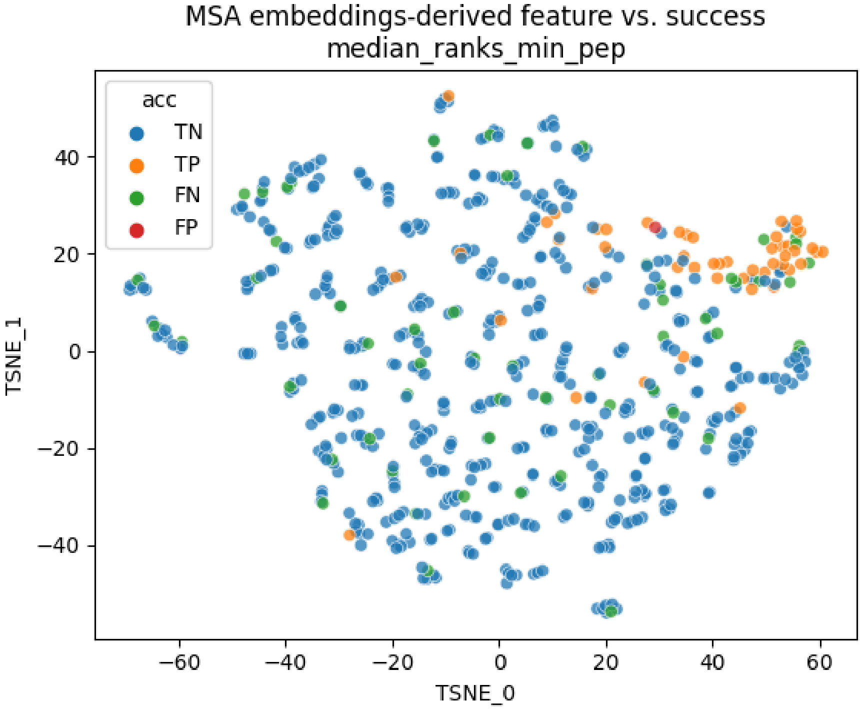


Figure 8: Minimum value of the “peptide part” of the MSA embeddings, taken the median over ranks, fed into a tSNE visualization.

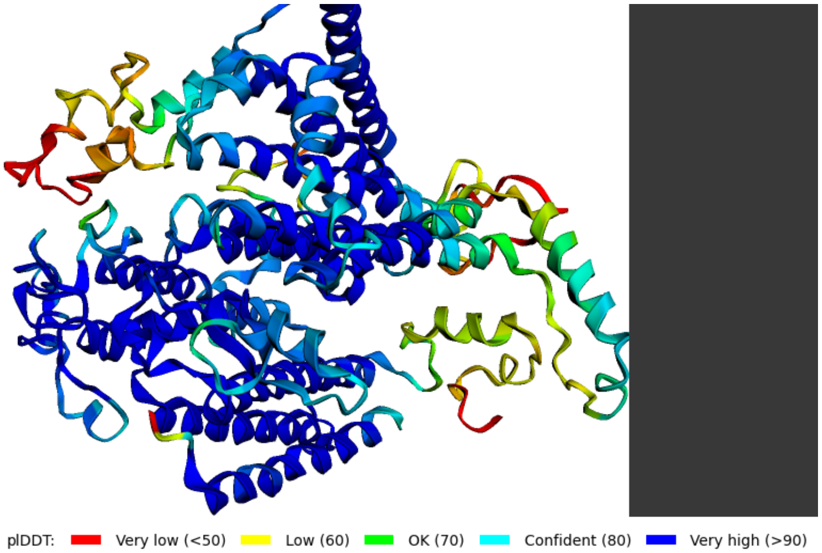


Figure 9: Best True Positive : 6k61_MB, multimer score of 0.93213. The peptide is the bottom-most helix, most of it and the neighboring protein parts being very high plDDT.

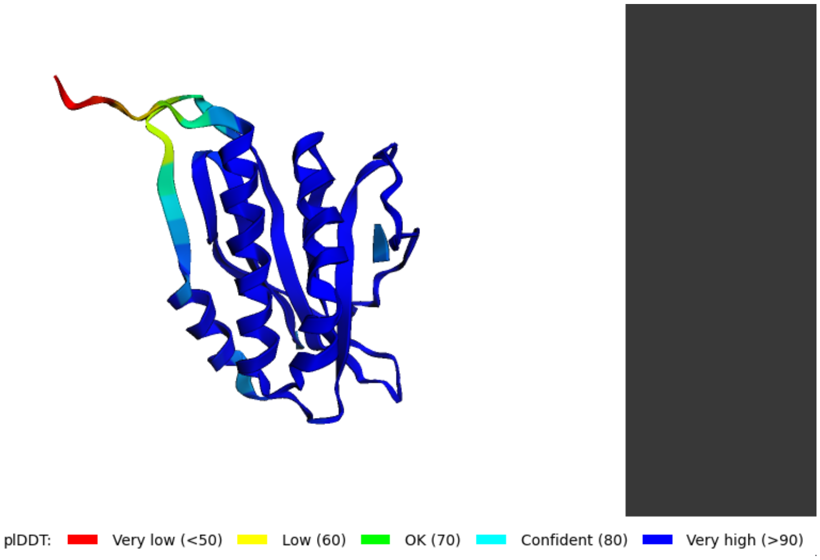


Figure 10: Worse False Positive : 6jfa_C6p8s_A, multimer score of 0.90723.The peptide is the tiny chain in the middle of the gap on the right, unfortunately with high plDDT, and the neighboring parts of the proteins very high too. As the negative labels are generated, this may be a biologically-sound complex.

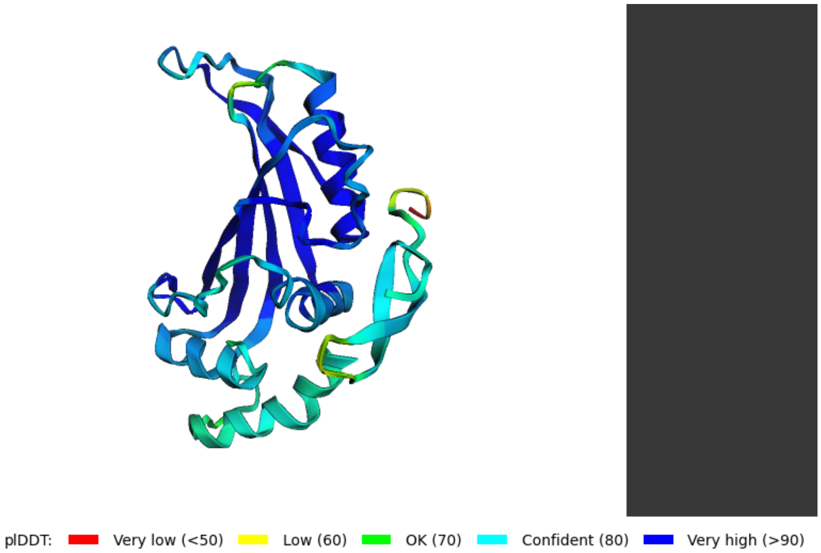


Figure 11: Worse False Negative : 6spb_1F, multimer score of 0.20899. The peptide is the loose helix-ish shape on the right, with no part reaching a very high plDDT.

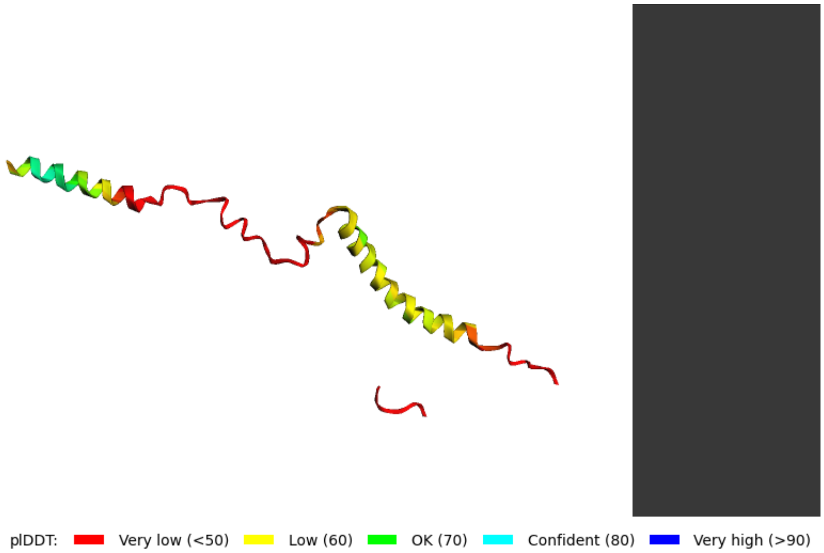


Figure 12: Best True Negative : 6pwc_B6jlj_E, multimer score of 0.07599. The success of predicting this complex as negative is questionable as neither object seems correctly predicted on their own at all.

5 Discussion

It is important to keep in mind these findings are from a single, fairly small dataset that aims to be an overview of the global Peptide - Protein landscape, and the behaviors observed and conclusions reached may not generalize to either general PPI datasets or datasets focused on specific areas. Additionally, as there is only one complex per ECOD family in the dataset, a data point that works particularly well here may not cause the entire family it is a representative of to work as well.

It stands to reason that the performance increase between Wallner's work with the previous versions of AF-M and this one may have consequences, particularly due to the reduction of sampling from 200 to only 2 seeds, for a 100x reduction in time and memory weight per complex assuming same runtime and weight per prediction, which should have both reduced. This should allow running such an interaction prediction process in many more situations where a longer, heavier process wasn't viable, such as screening over entire databases of complexes.

Other consequences not studied in this work that may demand future attention are whether this increase of performance from AF-M v2 to v3 on the interaction prediction task also comes with a similar upgrade on contact prediction, overall structure prediction, interface bonds prediction, or other similar related tasks.

Since the best single feature is based on the `multimer` score, which is a composition of the pTM and ipTM scores yet rank better than each, further study could find a better composition ratio than the default 80% iptm/20% ptm used here. Likewise, it could be studied whether taking the median over all ranks is really ideal or if other percentiles could work better.

The MSA embeddings showing a difference for the fake "peptide part" and its protein counterpart that I cut by simply re-applying the input sequence's structure is definitely a surprise, and a mystery by my current understanding of the AlphaFold pipeline. If the input sequences are not re-injected after the input in the architecture, the embeddings layers that lie at least 100 neuron layers (all types regrouped) deep should be near-independant of the structure of the complex's sequence.

It is possible in other datasets or other data populations that the MSA embeddings, and the Pairwise embeddings unused in this work, serve as the most informative AlphaFold outputs to be used as input to a downstream task, if fed into large models. This seems however redundant and undesirable on this dataset considering the performance reached by the single features.

As to the ML models, it is a bit disappointing that they don't surpass any of the few high-performing features they are given as inputs, though the models have been comparatively much less explored, as only their decision threshold has been optimized for the task. It remains possible that one of these ML classifiers reach higher performance by modifying their parameters towards aligning with the 5:1 Precision-Recall objective function, for those that have the ability in their design to do so. Naturally, hyperparameters optimization will also increase the performance.

6 Conclusion

This work shows that both confidence-based and distance-based features have high potential for predicting Peptide - Protein Interactions based on 2 seeds of AlphaFold-Multimer v3. In particular, the method used by Wallner for CASP15 (taking the median over all ranks of the multimer score) comes first amongst 161 features and a few classic ML classifiers, with the best compromise reached of 98% Precision/48% Recall.

This is a massive upgrade over Wallner's work which samples 200 seeds and ensembles both versions 1 and 2 of AlphaFold-Multimer and another tool, and reaches Precision/Recall compromises of 100/15, 95/20, 80/40%. Such a performance increase calls for further study of other possible similar upgrades in other similar tasks.

Studying the MSA embeddings for their possible use for data viz or clustering reveal that the embeddings still carry a notable difference in the Peptide side and Protein side of the sequence-analog dimension, though they are placed at the near end of the AlphaFold pipeline.

References

[1] Jumper, J., Evans, R., Pritzel, A. et al. Highly accurate protein structure prediction with AlphaFold. Nature 596, 583–589 (2021). https://doi.org/10.1038/s41586-021-03819-2

[2] Kryshtafovych, A, Schwede, T, Topf, M, Fidelis, K, Moult, J. Critical assessment of methods of protein structure prediction (CASP)—Round XIV. Proteins. 2021; 89(12): 1607- 1617. doi:10.1002/prot.26237

[3] Yoshitaka Moriwaki (@Ag_smith). Twitter post: AlphaFold2 can also predict heterocomplexes. all you have to do is input the two sequences you want to predict and connect them with a long linker. https://twitter.com/Ag_smith/status/1417063635000598528. 2021-07-19.

[4] Minkyung Baek (@minkbaek). Twitter post: Adding a big enough number for residue_index feature is enough to model hetero-complex using AlphaFold (green&cyan: crystal structure / magenta: predicted model w/ residue_index modification). https://twitter.com/minkbaek/status/1417538291709071362. 2021-07-20.

[5] Evans R et al. Protein complex prediction with AlphaFold-Multimer. bioRxiv 2021.10.04.463034; doi: https://doi.org/10.1101/2021.10.04.463034

[6] DeepMind's AlphaFold repository. AlphaFold v2.1.0 : Nov 2, 2021. AlphaFold v2.2.0 : Mar 10, 2022. AlphaFold v2.3.0 : Dec 13, 2022. https://github.com/deepmind/alphafold/releases

[7] Mirdita, M., Schütze, K., Moriwaki, Y. et al. ColabFold: making protein folding accessible to all. Nat Methods 19, 679–682 (2022). https://doi.org/10.1038/s41592-022-01488-1

[8] Steinegger, M., Söding, J. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. Nat Biotechnol 35, 1026–1028 (2017). https://doi.org/10.1038/nbt.3988

[9] Johansson-Åkhe I, Wallner B. Improving peptide-protein docking with AlphaFold-Multimer using forced sampling. Front Bioinform. 2022 Sep 26;2:959160. doi: 10.3389/fbinf.2022.959160. PMID: 36304330; PMCID: PMC9580857.

[10] Wallner B. AFsample: Improving Multimer Prediction with AlphaFold using Aggressive Sampling. bioRxiv 2022.12.20.521205; doi: https://doi.org/10.1101/2022.12.20.521205

[11] CASP. "CASP15 Bjorn Wallner" YouTube, 18 Jan 2023, https://youtu.be/fomZv3SYnz8

[12] Gao, M., Nakajima An, D., Parks, J.M. et al. AF2Complex predicts direct physical interactions in multimeric proteins with deep learning. Nat Commun 13, 1744 (2022). https://doi.org/10.1038/s41467-022-29394-2

[13] Humphreys, I.R., Pei, J., Baek, M. et al. Structures of core eukaryotic protein complexes. bioRxiv 2021.09.30.462231; doi: https://doi.org/10.1101/2021.09.30.462231

[14] Billings, W.M., Morris, C.J. & Della Corte, D. The whole is greater than its parts: ensembling improves protein contact prediction. Sci Rep 11, 8039 (2021). `https://doi.org/10.1038/s41598-021-87524-0`

[15] Wen Z, He J, Tao H, Huang SY. PepBDB: a comprehensive structural database of biological peptide-protein interactions. Bioinformatics. 2019 Jan 1;35(1):175-177. doi: 10.1093/bioinformatics/bty579. PMID: 29982280.

[16] Lei, Y., Li, S., Liu, Z. et al. A deep-learning framework for multi-level peptide–protein interaction prediction. Nat Commun 12, 5465 (2021). `https://doi.org/10.1038/s41467-021-25772-4`

[17] Cheng H, Schaeffer RD, Liao Y, Kinch LN, Pei J, Shi S, Kim BH, Grishin NV. ECOD: an evolutionary classification of protein domains. PLoS Comput Biol. 2014 Dec 4;10(12):e1003926. doi: 10.1371/journal.pcbi.1003926. PMID: 25474468; PMCID: PMC4256011.