ColabFold - Making protein folding accessible to all

Milot Mirdita,[1,*] Konstantin Schütze,[2] Yoshitaka Moriwaki,[3,4] Lim Heo,[5] Sergey Ovchinnikov,[6,7,*] and Martin Steinegger[2,8,*]

ColabFold offers accelerated protein structure and complex predictions by combining the fast homology search of MMseqs2 with AlphaFold2 or RoseTTAFold. ColabFold's $20-30$x faster search and optimized model use allows predicting thousands of proteins per day on a server with one GPU. Coupled with Google Colaboratory, ColabFold becomes a free and accessible platform for protein folding. ColabFold is open-source software available at github.com/sokrypton/ColabFold. Its novel environmental databases are available at colabfold.mmseqs.com
Contact: milot.mirdita@mpibpc.mpg.de, so@fas.harvard.edu, martin.steinegger@snu.ac.kr

Predicting the three-dimensional structure of a protein from its sequence alone remains an unsolved problem. However, by exploiting the information in multiple sequence alignments (MSAs) of related proteins as raw input features for end-to-end training, AlphaFold2 [1] was able to predict the 3D atomic coordinates of folded protein structures at an median GDT-TS of 92.4% in the latest CASP14 [2] competition. The accuracy of many of the predicted structures was within the error margin of experimental structure determination methods. Many ideas of AlphaFold2 were independently reproduced and implemented in RoseTTAFold [3]. Additionally to single chain predictions, RoseTTAFold was shown to model protein complexes. Evans *et al.* [3] also announced a refined version of AlphaFold2 for complex prediction. Thus, two highly accurate open-source prediction methods are now publicly available.

In order to leverage the power of these methods researchers require powerful compute-capabilities. First, to build diverse MSAs, large collections of protein sequences from public reference [4] and environmental [1, 5] databases are searched using the most sensitive homology detection methods HMMer [6] and HHblits [7]. Due to the large database sizes these searches can take up to hours for a single protein, while requiring over two terabyte of storage space alone. Second, to execute the deep neural networks GPUs with a large amount of GPU RAM are required even for relatively common protein sizes of ∼1000 residues. Though, for these the MSA generation dominates the overall run-time (**Supplementary Fig. 1**).

To enable researchers without these resources to use AlphaFold2 independent solutions based on Google Colaboratory were developed. Colaboratory is a proprietary version of Jupyter Notebook hosted by Google. It is accessible for free to logged-in users and includes access to powerful GPUs. Tunyasuvunakool *et al.* [8] developed an AlphaFold2 Jupyter Notebook for Google Colaboratory (referred to as AlphaFold-Colab), where the input MSA is built by searching with HMMer against a clustered UniProt and an eight-fold reduced environmental databases. Resulting in less accurate predictions, while still requiring long search times.

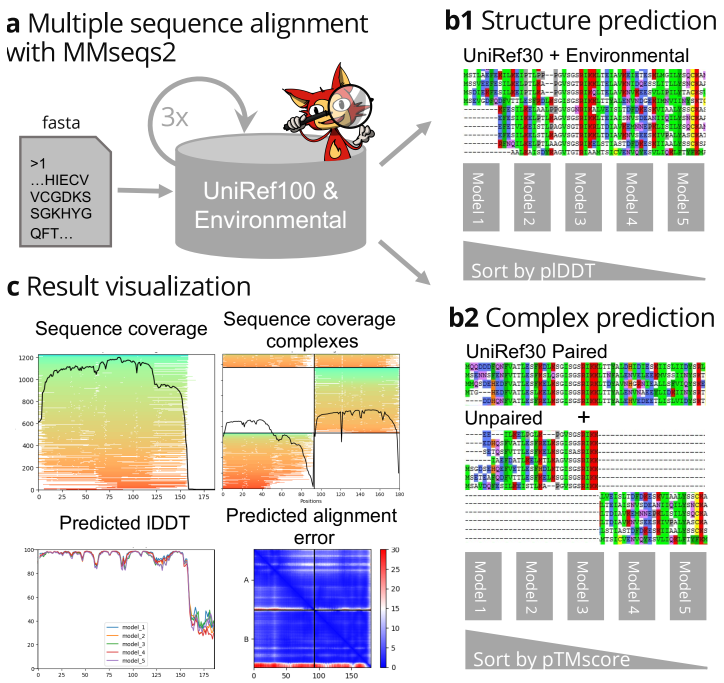

FIG. 1. (**a**) ColabFold sends a FASTA input sequence to a MMseqs2 server searching two databases UniRef100 and a database of environmental sequences with three profile-search iterations each. The second database is searched using a sequence-profile generated from the UniRef100 search as input. The server generates two MSAs in A3M format containing all detected sequences. (**b1**) For single structure predictions we filter both A3Ms using a diversity aware filter and return this to be provided as the MSA input feature to the AlphaFold2 models. (**b2**) For complex prediction we pair the top hits within the same species to resolve the inter-complex contacts and additionally add two unpaired MSAs (same to **b1**) to guide the structure prediction. (**c**) To help researchers judge the prediction quality we visualize MSA depth and diversity and show the AlphaFold2 confidence measures (pLDDT and PAE).

Here, we present ColabFold, a fast and easy to use software for protein structure and homo- and heteromer complex prediction, for use as a Jupyter Notebook inside Google Colaboratory, on researchers' local computers as a notebook or through a command line interface. ColabFold speed-ups the prediction by replacing the AlphaFold2's input feature generation stage with a fast MMseqs2 [9, 10] search. It additionally implements speed-ups for predictions of multiple structures by avoiding recompilation and adding early stop criteria. We show that ColabFold outperforms AlphaFold-Colab and matches AlphaFold2 on CASP14 targets while being 20-30 times faster. ColabFold can compute a proteome (excluding proteins >1000 residues) in 41 hours on a consumer GPU.

[1] Quantitative and Computational Biology, Max Planck Institute for Biophysical Chemistry, Göttingen, Germany. [2] School of Biological Sciences, Seoul National University, Seoul, South Korea. [3] Department of Biotechnology, Graduate School of Agricultural and Life Sciences, The University of Tokyo, Tokyo, Japan. [4] Collaborative Research Institute for Innovative Microbiology, The University of Tokyo, Tokyo, Japan. [5] Department of Biochemistry and Molecular Biology, Michigan State University, East Lansing, MI 48824, USA. [6] JHDSF Program, Harvard University, Cambridge, MA 02138, USA. [7] FAS Division of Science, Harvard University, Cambridge, MA 02138, USA. [8] Artificial Intelligence Institute, Seoul National University, Seoul, South Korea [*] These authors contributed equally and are ordered alphabetically.

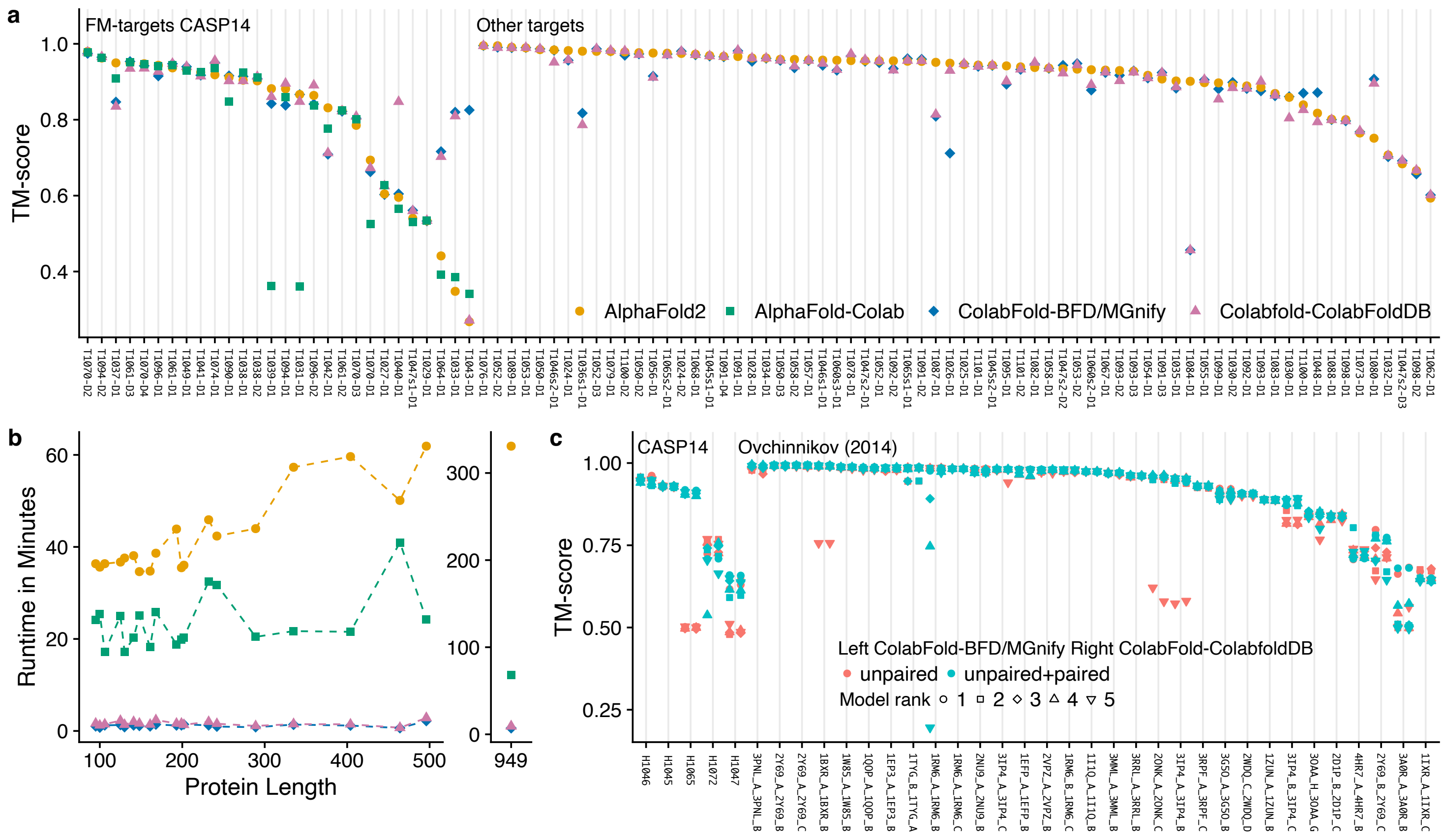

FIG. 2. (**a**) Structure prediction comparison of AlphaFold2 (yellow), AlphFold-Colab (green) and ColabFold with BFD/MGnify (blue) and with the ColabFoldDB (magenta) using predictions of 96 domains of 69 CASP14 targets. The 28 domains from the 20 free-modeling (FM) targets are shown first. FM targets were used to optimize MMseqs2 search parameters. Each target was evaluated for each individual domain (in total 96 domains). (**b**) MSA generation time for each CASP14 FM target sorted by protein length (same colors as before). FM target T1064 shown separately to improve readability. (**c**) Comparison of ColabFold complex predictions with unpaired (red) and unpaired+paired (blue) MSA-pairing modes, the databases BFD/MGnify (left of line) and ColabFoldDB (right). See **Supplementary Fig. 2** for comparison to paired-only mode.

ColabFold (**Fig. 1**) consists of three parts: (1) An MMseqs2 based homology search server to build diverse MSAs and to find templates. The server efficiently aligns input sequence(s) against the UniRef100, the PDB70 and an environmental sequence set. (2) A Python library that communicates with the MMseqs2 search server, prepares the input features for (single or complex) structure inference, and visualizes of results. This library also implements a command line interface. (3) Jupyter notebooks for basic, advanced and batch use (Methods "ColabFold notebooks") using the Python library.

In ColabFold we replace the sensitive search methods HMMer and HHblits by MMseqs2. We optimized the MSA generation by MMseqs2 to have the following three properties: (1) MSA generation should be fast. (2) The MSA has to capture diversity well and (3) it has to be small enough to run on GPUs with limited RAM. Reducing the memory requirement is especially helpful in Google Colaboratory where the provided GPU is selected from a pool with widely differing capabilities. While (1) is achieved through the fast MMseqs2 prefilter for (2 and 3) we developed a search workflow to maximize sensitivity (Methods "MSA generation") and a new filter that samples the sequence space evenly (Methods "New diversity aware filter" and **Supplementary Fig. 3**). Prediction quality highly depends on the input MSA. However, often only a few (∼30) sufficiently diverse sequences are enough to produce high quality predictions [1].

Additionally, we combined the BFD and MGnify databases that are used in AlphaFold2 by HHblits and HMMer respectively into a combined redundancy reduced version we refer to as BFD/MGnify (Methods "Reducing size of BFD/MGnify"). The environmental search database presented an opportunity to improve structure predictions of non-bacterial sequences, as e.g., eukaryotic protein diversity is not well represented in the BFD and MGnify databases. Limitations in assembly and gene calling due to complex intron/exon structures result in under representation in reference databases. We therefore extended the BFD/MGnify with additional metagenomic protein catalogues containing eukaryotic proteins [11, 12, 13], phage catalogues [14, 15] and an updated version of MetaClust [16]. We refer to this database as ColabFoldDB (Methods "ColabFoldDB"). In **Supplementary Fig. 4** we show that the ColabFoldDB in comparison to the BFD/MGnify produces more diverse MSAs for PFAM [17] domains with <30 members.

To compare the accuracy of predicted structures we compared AlphaFold2 (default settings with templates), AlphaFold-Colab (no templates), and ColabFold (no templates) with the BFD/MGnify and ColabFoldDB on TM-scores for all targets from the CASP14 competition (**Fig. 2a**), split by free modeling (FM) targets on the left and the remaining ones on the right. We show this split as we used the FM-targets for optimization of search workflow parameters.

The mean TM-scores for the FM targets are 0.826, 0.818, 0.79 and 0.744 for ColabFold (BFD/MGnify), ColabFold (ColabFoldDB), AlphaFold2 and the AlphaFold-Colab, respectively. Over all CASP14 targets the TM-scores are 0.88, 0.877 and 0.88 for the former three respectively. For AlphaFold-Colab we measured TM-scores only for FM targets as it cannot be used stand-alone.

ColabFold could not predict T1084 well as MMseqs2 suppresses all databases hits as false positives due to its amino acid composition filter and masking procedure. If these filters are deactivated T1084 can be predicted with an TM-score of 0.872 (**Supplementary Fig. 5**).

ColabFold is on average 5x faster for single predictions than AlphaFold2 and AlphaFold-Colab, when taking both MSA generation (**Fig. 2b**) and model inference into account.

AlphaFold2 itself has no capabilities to model complexes. However, we found that by combining two sequences with a glycine linker [18] it could often successfully model complexes. Shortly afterwards, Baek [19] found that incrementing the model-internal residue index - the method that was used in RoseTTAFold - could also be used in AlphaFold2.

For high quality predictions it was shown that sequences should be provided in paired-form to AlphaFold2 [20]. We implemented a similar pairing procedure (Methods "MSA pairing for complex prediction") and show the complex prediction capabilities of ColabFold in **Fig. 2c**. We achieve high accuracy in complex prediction in two datasets from Ovchinnikov *et al.* [21] and the CASP14 protein complex targets with two unique sequences (Methods "Complex Benchmark" for benchmark details). We note though that the structures from [21] were already public and were likely used as individual chains during the training of AlphaFold2.

Fig. 3 shows two examples of ColabFold's complex prediction capabilities: (**a**) shows a homo-six-mer and (**b**) shows a D-methionine transport system composed of three different proteins. For single structure prediction AlphaFold2 provides a pLDDT measure to indicate the prediction quality. A high pLDDT does not necessarily indicate a correct complex prediction, though the inter-complex predicted alignment error (PAE) helps to rank complexes. We visualize plots of PAE and complex conformation to help users judge the prediction quality of a complex. An example for heteromer complex prediction is shown in **Supplementary Fig. 6** with its PAE plot. Furthermore, ColabFold complexes were successfully used to aid the cryo-EM structure determination of the 120 MDa human nucleopore complex [22].

In ColabFold we expose many internal parameters of AlphaFold2 to aid users to model difficult targets, such as the recycle count (default 3). It controls the number of times the prediction is repeatedly feed through the model. For difficult targets as well as for designed proteins without known homologs additional recycling iterations can result in a high quality prediction (**Supplementary Fig. 7**).

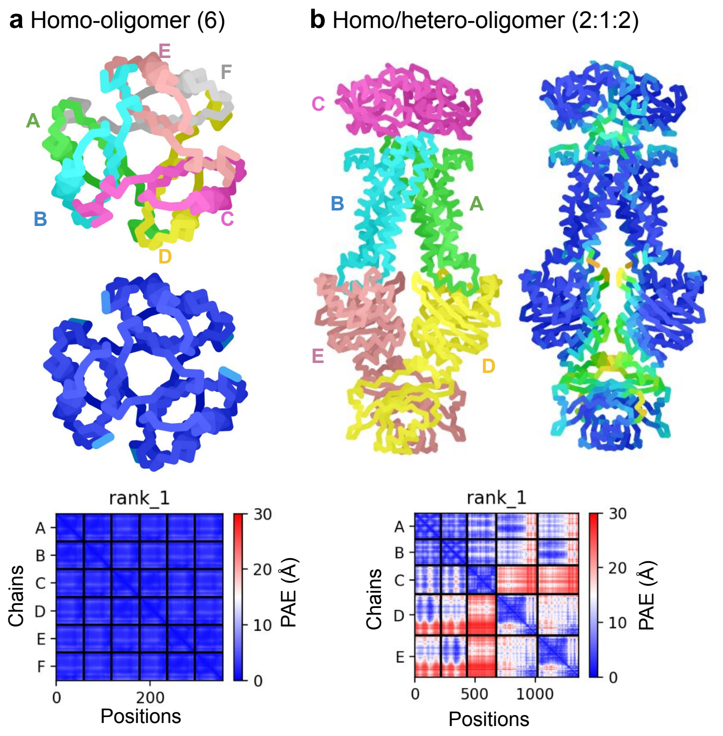

FIG. 3. Anecdotal examples showcasing the capabilities of advanced ColabFold features. (**a**) Setting the homo-oligomer setting to 6, allows modeling of the homo-6-mer structure of 4-Oxalocrotonate Tautomerase. Colored by chain (top), pLDDT (predicted Local Distance Difference Test, bottom). The inter PAE (Predicted Aligned Error) between chains is very low indicating a confident prediction. (**b**) Providing three different proteins with 2:1:2 homo-oligomer setting allows modeling a hetero-complex with mismatching symmetries of the D-methionine transport system.

To meet the demand for high throughput structure prediction we introduced several features in ColabFold. (1) MSA generation can be executed in batch-mode independently from model batch-inference. (2) We compile only two of the five AlphaFold2 models and reuse weights. (3) We provide a batch execution mode, that avoids recompilation for sequences of similar length. (4) We implement early stop criteria, to avoid running additional recycles or models if a sufficiently accurate structure was already found. All together, we show that the proteome of 1762 proteins shorter than 1000 aa of the archaeon *Methanocaldococcus jannaschii* can be predicted in 40h on one Nvidia RTX 3090 (Methods "Proteome Benchmark").

ColabFold builds beyond the initial offerings of Alphafold2 by improving its sequence search, providing tools for modeling homo- and heteromer complexes, exposing advanced functionality, expanding the environmental databases and performing structure prediction in batch within a minute.

In summary, ColabFold makes high quality protein structure prediction accessible and additionally provides novel features to explore the full potential of AlphaFold2 and RoseTTAFold.

REFERENCES

[1] Jumper, J. *et al.* *Nature* **596**, 583–589 (2021).
[2] Kryshtafovych, A. *et al.* *Proteins* 1–11 (2021).
[3] Evans, R. *et al.* *bioRxiv* 2021.10.04.463034 (2021).
[4] UniProt Consortium. *Nucleic Acids Res.* **47**, D506–D515 (2019).
[5] Mitchell, A. L. *et al.* *Nucleic Acids Res.* **48**, D570–D578 (2020).
[6] Eddy, S. R. *PLoS Comput. Biol.* **7**, e1002195 (2011).
[7] Steinegger, M. *et al.* *BMC Bioinform.* **20**, 473 (2019).
[8] Tunyasuvunakool, K. *et al.* *Nature* **596**, 590–596 (2021).
[9] Steinegger, M. & Söding, J. *Nat. Biotechnol.* **35**, 1026–1028 (2017).
[10] Mirdita, M. *et al.* *Bioinformatics* **35**, 2856–2858 (2019).
[11] Levy Karin, E. *et al.* *Microbiome* **8**, 48 (2020).
[12] Delmont, T. O. *et al.* *bioRxiv* 2020.10.15.341214 (2020).
[13] Alexander, H. *et al.* *bioRxiv* 2021.07.25.453713 (2021).
[14] Nayfach, S. *et al.* *Nat. Microbiol.* **6**, 960–970 (2021).
[15] Camarillo-Guerrero, L. F. *et al.* *Cell* **184**, 1098–1109.e9 (2021).
[16] Steinegger, M. & Söding, J. *Nat. Commun.* **9**, 2542 (2018).
[17] Mistry, J. *et al.* *Nucleic Acids Res.* **49** (2021).
[18] Moriwaki, Y. 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).
[19] Baek, M. 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).
[20] Bryant, P. *et al.* *bioRxiv* 2021.09.15.460468 (2021).
[21] Ovchinnikov, S. *et al.* *eLife* **3**, e02030 (2014).
[22] Mosalaganti, S. *et al.* *bioRxiv* 2021.10.26.465776 (2021).

ACKNOWLEDGEMENTS

We thank Johannes Söding for providing computational resources. John Jumper and Tim Green for answering questions regarding AF2. Minkyung Baek for the complex residue trick. Do-Yoon Kim for creating the ColabFold logo. Enzo Guerrero-Araya and Jakub Kaczmarzyk for providing bug fixes. Alon Markovich and Julia Varga for notifying us about MSA quality issues. Harriet Alexander for providing the TOPAZ proteins as a single file to download. We thank all users for using ColabFold and reporting issues.

This work used the Scientific Compute Cluster at GWDG, the joint data center of Max Planck Society for the Advancement of Science (MPG) and University of Göttingen. Milot Mirdita acknowledges the BMBF CompLifeSci project horizontal4meta. Martin Steinegger acknowledges support from the National Research Foundation of Korea grant [2019R1A6A1A10073437, 2020M3A9G7103933, 2021R1C1C102065]; New Faculty Startup Fund and the Creative-Pioneering Researchers Program through Seoul National University. Yoshitaka Moriwaki acknowledges support from Platform Project for Supporting Drug Discovery and Life Science Research (Basis for Supporting Innovative Drug Discovery and Life Science Research (BINDS)) from AMED under Grant Number JP21am0101107. For this project, Sergey Ovchinnikov was supported by the National Science Foundation under Grant No. MCB2032259. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

AUTHOR CONTRIBUTION

M.M., K.S., S.O. and M.S. performed research and programming, M.M., S.O. and M.S. jointly designed the research and wrote the manuscript. Y.M. provided the initial methodology for hetero-complex modeling and created an installer for use on local servers. L.H. provided initial benchmarking.

COMPETING INTERESTS

The authors declare no competing interests.

MATERIALS AND METHODS

ColabFold notebooks ColabFold has four main Jupyter notebooks [23]: `AlphaFold2_mmseqs2` for basic use that supports protein structure prediction using (1) MSAs generated by MMseqs2, (2) custom MSA upload, (3) using template information, (4) relaxing the predicted structures using amber force fields [24], and (5) monomer complex prediction. `AlphaFold2_advanced` for advanced users additionally supports (6) MSA generation using HMMer (same as AlphaFold-Colab), (7) the sampling of diverse structures by iterating through a series of random seeds (`num_samples`), and (8) control of AlphaFold2 model internals, such as changing the number of recycles (`max_recycle`), number of ensembles (`num_ensemble`), and enabling the stochastic part of the models via the (`is_training`) option. `AlphaFold2_batch` for batch prediction of multiple sequences or MSAs. The batch notebook saves time by avoiding recompilation of the AlphaFold2 models ("Avoid recompiling during batch computation") for each individual input sequence. `RoseTTAFold` for basic use of RoseTTAFold that supports protein structure prediction using (1) MSAs generated by MMseqs2, (2) custom MSAs and (4) sidechain prediction using SCWRL4 [25].

ColabFold command line interface We initially focused on making ColabFold as widely available as possible through our Notebooks running in Google Colaboratory. To meet the demand for a version that runs on local users' machines, we released "LocalColabFold". LocalColabFold can take command line arguments to specify an input FASTA file, an output directory, and various options to tweak structure predictions. LocalColabFold runs on wide range of operating systems, such as Windows 10 or later (using Windows Subsystem for Linux 2), macOS, and Linux. The structure inference and energy minimization are accelerated if a CUDA 11.1 or later compatible GPU is present. LocalColabFold is available as free open-source software at `github.com/YoshitakaMo/localcolabfold`.

Specifically for running large numbers of protein complexes or structure predictions e.g., for an entire proteome (Methods "Proteome benchmark"), we provide the `colabfold_batch` command line tool through the `colabfold` python package. It can be installed with `pip install colabfold`, followed by `pip install -U "jax[cuda]" -f https://storage.googleapis.com/jax-releases/jax_releases.html`. It can be used as `colabfold_batch input_file_or_directory output_directory`, supporting FASTA, A3M and CSV files as input.

MSA generation by MMseqs2 ColabFold sends the query sequence to a MMseqs2 server [12]. It searches the sequence(s) with three iterations against the consensus sequences of the UniRef30, a clustered version of the UniRef100 [26]. We accept hits with an E-value of lower than 0.1. For each hit, we realign its respective UniRef100 cluster member using the profile generated by the last iterative search, filter them (Methods "New diversity aware filter") and add these to the MSA. This expanding search results in a speed up of ~10x as only 29.3 million cluster consensus sequence are searched instead of all 277.5 million UniRef100 sequences. Additionally, it has the advantages to be more sensitive since the cluster consensus sequences are used. We use the UniRef30 sequence-profile to perform an iterative search against the BFD/MGnify or ColabFoldDB using the same parameters, filters and expansion strategy.

New diversity aware filter To limit the number of hits in the final MSA we use the HHblits diversity filtering algorithm [8] implemented in MMseqs2 in multiple stages: (1) During UniRef cluster expansion, we filter each individual UniRef30 cluster before adding the cluster members to the MSA, such that no cluster-pair has a higher maximum sequence identity than 95% (`--max-seq-id 0.95`. (2) After realignment enable only the `--qsc 0.8` threshold and disable all other thresholds (`--qid 0 --diff 0 --max-seq-id 1.0`). Additionally, the qsc filtering is only used if least 100 hits were found (`--filter-min-enable 100`). (3) During MSA construction we filter again with the following parameters: `--filter-min-enable 1000 --diff 3000 --qid 0.0,0.2,0.4,0.6,0.8,1.0 --qsc 0 --max-seq-id 0.95`. Here, we extended the HHblits filtering algorithm to filter within a given sequence identity bucket, such that it cannot eliminate redundancy across filter buckets. Our filter keeps the 3000 most diverse sequences in the identity buckets]0.0-0.2],]0.2-0.4],]0.4-0.6],]0.6-0.8] and]0.8-1.0]. In buckets containing less than 1000 hits we disable the filtering.

New MMseqs2 pre-computed index to support expanding cluster members MMseqs2 was initially built to perform fast many-against-many sequence searches. Mirdita *et al.* [11] improved it to also support fast single-against-many searches. This type of search requires the database to be index and stored in memory. `mmseqs createindex` indexes the sequences and stores all time-consuming-to-compute data structures used for MMseqs2 searches to disk. We load the index into the operating systems cache using *vmtouch* (`github.com/hoytech/vmtouch`) to allow calls to the different MMseqs2 modules become near-overhead free. We extended the index to store, in addition to the already present cluster consensus sequences, all member sequences and the pairwise alignments of the cluster representatives to the cluster members. With these resident in cache, we eliminate the overhead of the remaining module calls.

Reducing size of BFD/MGnify To keep all required sequences and data structures in memory we needed to reduce the size of the environmental databases BFD and MGnify, as both databases together would have required ~517 GB RAM for headers and sequences alone.

BFD is a clustered protein database consisting of ~2.2 billion proteins organized in 64 million clusters. MGnify (2019_05) contains ~300 million environmental proteins. We merged both databases by searching the MGnify sequences against the BFD cluster representative sequences using MMseqs2. Each MGnify sequence with a sequence identity of >30% and a local alignment that covers at least 90% of its length is assigned to the respective BFD cluster. All unassigned sequences are clustered at 30% sequence identity and 90% coverage (`--min-seq-id 0.3 -c 0.3 --cov-mode 1 -s 3`) and merged with the BFD clusters, resulting in 182 million clusters. In order to reduce the size of the database we filtered each cluster keeping only the 10 most diverse sequences using (`mmseqs filterresult --diff 10`). This reduced the total number of sequences from 2.5 billion to 513 million, thus requiring only 84 GB RAM for headers and sequences.

ColabFoldDB We built ColabFoldDB by expanding the BFD/MGnify with metagenomic sequences from various environments. To update the database, we searched the proteins from the SMAG (eukaryotes) [14], MetaEuk (eukaryotes) [13], TOPAZ (eukaryotes) [15], MGV (DNA viruses) [16], GPD (bacteriophages) [17] and updated version of MetaClust [17] against the BFD/MGnify centriods using MMseqs2 and assigned each sequence to the respective cluster if they have a 30% sequence identity at a 90% sequence overlap (`-c 0.9 -cov-mode 1 -min-seq-id 0.3`). All remaining sequences were clustered using MMseqs2 `cluster -c 0.9 -cov-mode 1 -min-seq-id 0.3` and appended to the database. We remove redundancy per cluster by keeping the most 10 diverse sequences using (`mmseqs filterresult --diff 10`). The final database consists of 209,335,865 million representative sequences and 738,695,580 members. See "Data availability" for input files. We extracted the MMseqs2 search workflow used in the server ("MSA generation by MMseqs2") into a standalone script `colabfold_search.sh` and provide it together with the databases.

Template information AlphaFold2 searches with HHsearch through a clustered version of the PDB (PDB70 [8]) to find the 20 top ranked templates. In order to save time, we use MMseqs2 [10] to search against the PDB70 cluster representatives as a prefiltering step to find candidate templates. This search is also done as part of the MMseqs2 API call on our server. Only the top 20 target templates according to E-value are then aligned by HHsearch. The accepted templates are given to AlphaFold2 as input features. This alignment step is done in the ColabFold client and therefore requires the subset of the PDB70 containing the respective HMMs. The PDB70 subset and the PDB mmCIF files are fetched from our server. For benchmarking, no templates are given to ColabFold.

Custom MSAs ColabFold allows researchers to upload their own MSAs. Any kind of alignment tool can be used to generate the MSA. The uploaded MSA can be provided in aligned FASTA, A3M, STOCKHOLM or Clustal format. We convert the respective MSA format into A3M format using the `reformat.pl` script from the HH-suite [8].

Modeling of protein-protein complexes Baek *et al.* [3] show that RoseTTAFold is able to model complexes, despite being trained only on single chains. This is done by providing a paired alignment and modifying the residue index. The residue index is used as an input to the models to compute positional embeddings. In AlphaFold2, we find the same to be true, although surprisingly the paired alignment is often not needed (**Fig. 2c**). AlphaFold2 uses relative positional encoding with a cap at $|i-j| \geq 32$. Meaning, any pair of residues separated by 32 or more are given the same relative positional encoding. By offsetting the residue index between two proteins to be > 32, AlphaFold2 treats them as separate poly-peptide chains. ColabFold integrates this for modeling complexes.

For homo-oligomeric complexes (**Fig. 3a**), the MSA is copied multiple times for each component. Interestingly, it was found that providing a separate MSA copy (padding by gap characters to extend to other copies) to work significantly better than concatenating left-to-right.

For hetero-oligomeric complexes (**Fig. 3b**), a separate MSA is generated for each component. The MSA is paired according to the chosen `pair_mode` ("MSA pairing for complex prediction"). Since pLDDT is only useful for assessing local structure confidence, we use the fine-tuned model parameters to return the PAE for each prediction. As illustrated in **Supplementary Fig. 6**, the inter-PAE (predicted aligned error) or the predicted TM-score (derived from PAE) can be used to rank and assess the confidence of the predicted protein-protein interaction.

MSA pairing for complex prediction A paired MSA helps AlphaFold2 to predict complexes more accurately only if orthologous genes are paired with each other. We followed a similar strategy as Bryant *et al.* [21] to pair sequences according to their taxonomic identifier. For the pairing we search each distinct sequence of a complex against the UniRef100 using the same procedure as described in "MSA generation". We return only hits that cover all complex proteins within one species and pair only the best hit (smallest e-value) with an alignment that covers the query to at least 50%. The pairing is implemented in the new MMseqs2 module `pairaln`.

For prokaryotic protein prediction, we additionally implemented the protocol described in [3] to pair sequences based on their distances in the genome as predicted from the UniProt accession numbers.

Taxonomic labels for MSA pairing To pair MSAs for complex prediction, we retrieve for each found UniRef100 member sequence the taxonomic identifier from the NCBI taxonomy [27]. The taxonomic labels are extracted from the lowest common ancestor field ("common taxon ID") of each UniRef100 sequence from the `uniref100.xml` (2021_03) file.

Complex benchmark We compare predictions of five CASP14 complex targets (H1045, H1046, H1047, H1065, H1072) and 32 targets from Ovchinnikov *et al.* [22] to their native structures using MM-align [28] and extract TM-scores. We used `colabfold_batch` with BFD/MGnify and ColabFoldDB to predict structures in three different modes: (1) without MSA pairing, (2) with MSA pairing as described in "MSA pairing for complex prediction" and (3) with MSA pairing and also adding unpaired sequences. Models are ranked by pTMscore predicted by AlphaFold2.

Avoid recompiling AlphaFold2 models The AlphaFold2 models are compiled using JAX [29] to optimize the model for specific MSA or template input sizes. When no templates are provided, we compile once and, during inference, replace the weights from the other models, using the configuration of model 5. This saves 7 minutes of compile time. When templates are enabled, model 1 is compiled and weights from model 2 are used, model 3 is compiled and weights from models 4 and 5 are used. This saves 5 minutes of compile time. If the user changes the sequence or settings, without changing the length or number of sequences in the MSA, the compiled models are reused without triggering recompilation.

Avoid recompiling during batch computation In order to avoid AlphaFold2 model recompilation for every protein AlphaFold2 provides a function to add padding to the input MSA and templates called `make_fixed_size`. However, this is not exposed in AlphaFold2. We used the function in our batch notebook as well as in our command line tool *colabfold_batch*, in order to maximize GPU utilization and minimize the need of model recompilation. We sort the input queries by sequence length and process them in ascending order. We pad the input features by 10% (by default). All sequences that lie within the query length and an additional 10% margin do not require to be recompiled, resulting in a large speed up for short proteins.
Speed-up of predictions through early stop AlphaFold2 computes five models. We noted that for prediction of high certainty (>85 pLDDT), all five models would often produce structures of very similar confidence. In order to speed up the computation we added a parameter to `colabfold_batch` to define an early stop criterion that halts additional model inferences if a given pLDDT or pTMscore threshold is reached.
Recycle count AlphaFold2 improves the predicted protein structure by recycling (by default) 3 times, meaning the prediction is fed multiple times through the model. We exposed the recycle count as a customizable parameter as additional recycles can often improve a model at the cost of a longer runtime. We also implemented an option to specify a tolerance threshold to stop early. For some designed proteins without known homologous sequences, this helped to fold the final protein (**Supplementary Fig. 7**).
Sampling of diverse structures To reduce memory requirements, only a subset of the MSA is used as input to the model. Alphafold2, depending on model configuration, subsamples the MSA to a maximum of 512 cluster centers and 1024 "extra" sequences. Changing the random seed can result in different cluster centers and thus different structure predictions. ColabFold provides an option to iterate through a series of random seeds, resulting in structure diversity. Further structure diversity can be generated by using the original or fine-tuned (`use_ptm`) model parameters and/or enabling (`is_training`) to activate the stochastic (dropout) part of model. Enabling the latter, can be used to sample an ensemble of models for the uncertain parts of the structure prediction.
Proteome benchmark We predict the proteome of the archaeon *Methanocaldococcus jannaschii*. Of the 1787 proteins we exclude the 25 proteins longer than 1000 residues, leaving 1762 proteins of 268 aa average length. We search in 58 min using 100 threads on a system with 2x64-core AMD EPYC 7742 CPUs and 2TB RAM using `colabfold_search.sh` against the ColabFoldDB ("ColabFoldDB"), though we reduce the sensitivity to the considerably faster `-s 6` setting. We then predict the structures on a single Nvidia RTX 3090 with 28 GB RAM in 39.6 h using only MSAs (no templates). For each query we stop early if any model reaches a pLDDT of at least 85. We extrapolate the runtime for no-early-stopping by multiplying the runtime of model 3 for each protein to five models, yielding an overall speedup of factor 2.8. We observe a high structural agreement with an median TM-Score of 0.986 and mean TM-score of 0.953 when comparing the best predictions of ColabFold and AlphaFold2 with TMalign [30].
Benchmark with CASP14 targets We compare the AlphaFold-Colab and the AlphaFold2 (commit `b88f8da`) against ColabFold (commit `2b49880`, **Fig. 2**) using all CASP14 [2] targets. ColabFold uses UniRef30 (2021_03) [31] and the BFD/Mgnify or ColabFoldDB. AlphaFold-Colab uses the UniRef90 (2021_03), MGnify (2019_05) and the small BFD. AlphaFold2 uses the `full_dbs` preset with and default databases downloaded with the `download_all_data.sh` script. The 69 targets contain 96 domains, among these are 20 FM-targets with 28 domains. We compared the predictions against the experimental structures using TMalign [30].
Measuring time for CASP14 and complex targets All ColabFold and AlphaFold2 benchmarks were executed on systems with 2x16 core Intel Gold 6242 CPUs with 192 GB RAM and 4x Nvidia Quadro RTX5000 GPUs. Only one GPU was used in each individual run.

ColabFold was executed using `colabfold_batch`. The MMseqs2 server which computes MSAs for ColabFold has 2x14 core Intel E5-2680v4 CPUs and 768 GB RAM. Each generated MSA was processed by a single CPU-core. Runtimes were computed from server logs.

Runtimes for AlphaFold2 were extracted from the `features` entry of generated `timings.json` file. Where indicated with multicore, AlphaFold2 was used with the default 8 CPU cores for HMMer and 4 CPU cores for HHblits to process one query. For a fair comparison, AlphaFold2 was modified to allow HMMer and HHblits to access one CPU core.

AlphaFold-Colab was executed in the browser using a Google Colab Pro account. Times for homology search were taken from the log output of the "Search against genetic databases" cell in the notebook. The JackHMMer search uses 8 threads.

DATA AVAILABILITY

ColabFold databases are available at
`colabfold.mmseqs.com`.
Input databases used for building ColabFold databases:
UniRef30: `uniclust.mmseqs.com`
BFD: `bfd.mmseqs.com`
MGnify: `ftp.ebi.ac.uk/pub/databases/metagenomics/peptide_database/2019_05`
PDB70: `wwwuser.gwdg.de/~compbiol/data/hhsuite/databases/hhsuite_dbs`
MetaEuk: `wwwuser.gwdg.de/~compbiol/metaeuk/2019_11/MetaEuk_preds_Tara_vs_euk_profiles_uniqs.fas.gz`
SMAG: `www.genoscope.cns.fr/tara/localdata/data/SMAGs-v1/SMAGs_v1_concat.faa.tar.gz`
TOPAZ: `osf.io/gm564`
MGV: `portal.nersc.gov/MGV/MGV_v1.0_2021_07_08/mgv_proteins.faa`
GPD: `ftp.ebi.ac.uk/pub/databases/metagenomics/genome_sets/gut_phage_database/GPD_proteome.faa`

REFERENCES

[23] Kluyver, T. *et al.* Jupyter notebooks - a publishing format for reproducible computsteinegger2018ational workflows. In *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, 87–90 (IOS Press, 2016).

[24] Eastman, P. *et al. PLoS Comput. Biol.* **13**, 1–17 (2017).

[25] Krivov, G. G. *et al. Proteins* **77**, 778795 (2009).

[26] Suzek, B. E. *et al. Bioinformatics* **31**, 926–932 (2015).

[27] Federhen, S. *Nucleic Acids Res.* **40**, D136–D143 (2012).

[28] Mukherjee, S. & Zhang, Y. *Nucleic Acids Res.* **37**, e83–e83 (2009).

[29] Bradbury, J. *et al.* JAX: composable transformations of Python+NumPy programs (2018).

[30] Zhang, Y. & Skolnick, J. *Nucleic Acids Res.* **33**, 2302–2309 (2005).

[31] Mirdita, M. *et al. Nucleic Acids Res.* **45**, D170–D176 (2017).