Gerador de proteínas para o tratamento da Hemofilia tipo B baseado em *Deep Reinforcement Learning*

Arthur Moriggi Pimenta

Dissertação apresentada ao
Instituto de Matemática e Estatística
da Universidade de São Paulo
para obtenção do título de
Mestre em Ciências

Programa: Ciência da Computação
Orientador: Prof. Dr. Roberto Hirata Junior

São Paulo
Novembro de 2023

Gerador de proteínas para o tratamento da Hemofilia tipo B baseado em *Deep Reinforcement Learning*

Arthur Moriggi Pimenta

Esta é a versão original da dissertação elaborada pelo candidato Arthur Moriggi Pimenta, tal como submetida à Comissão Julgadora.

Lista de abreviaturas

MSA	Alinhamento múltiplo de sequências (*Multiple Sequence Alignment*)
IS	Idêntidade de Sequência
TM-Score	Escores de Modelagem de Template (*Template Modeling Score*)
LLM	Modelo Grande de Linguagem (*Large Language Model*)
PPO	Otimização de Política Proxima (*Proximal Policy Optimization*)
IME	Instituto de Matemática e Estatística
USP	Universidade de São Paulo

Lista de figuras

Lista de tabelas

Sumário

Introdução

Em andamento

A hemofilia é uma doença hereditária rara, caracterizada por uma deficiência nos fatores de coagulação sanguínea. Existem dois principais tipos de hemofilia: a A, causada pela deficiência do fator VIII de coagulação, e a hemofilia B, causada pela deficiência do fator IX. Os pacientes com hemofilia enfrentam um risco maior de sofrer com hemorragias graves, tanto internas quanto externas, podendo levar a complicações debilitantes e até mesmo à morte PM, 2020.

Embora tenham sido obtidos avanços notáveis no tratamento da doença nas últimas décadas, muitos desafios ainda persistem Gouw, 2013. O tratamento tradicional da hemofilia envolve a infusão de fatores de coagulação recombinantes ou derivados do plasma sanguíneo Gouw, 2013. Apesar de sua eficácia na prevenção de hemorragias, esse tratamento possui limitações, tais como a necessidade de infusões frequentes, a possibilidade de desenvolvimento de inibidores anticoagulantes e, principalmente, os altos custos envolvidos Mancuso, 2005. Pegar algum artigo que fale sobre a instabilidade do fator IX, que faz com que o tratamento seja caro e sobre a resposta imune. O projeto de sequências proteicas surge como uma abordagem que tem potencial de reduzir substancialmente os custos associados ao tratamento da doença, ao criar proteínas sob medida com eficácia comparável.

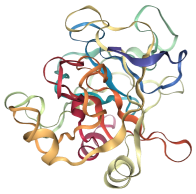
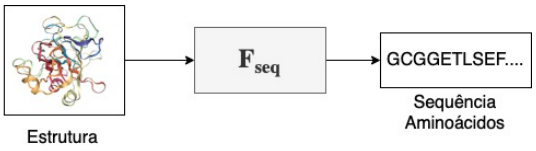


Figura 1: *Estrutura Alvo: Fator IX - Hemofilia tipo B*

0.1 Projeto de sequências

O projeto de sequências se refere ao processo de criar ou otimizar uma sequência de aminoácidos de modo a produzir uma proteína cuja estrutura tridimensional possua características desejadas. Esta é uma tarefa desafiadora que, a depender do tamanho da sequência, é inviável de se resolver através de uma busca por força bruta, devido as inúmeras permutações possíveis que se pode obter com os 20 aminoácidos conhecidos na natureza Wenze Ding e Gong, 2022.



O projeto de sequências se resume a desenvolver uma função F_{seq} que mapeia uma estrutura protéica tridimensional em uma sequência de aminoácidos.

Figura 2: *Projeto de Sequências*

0.1.1 Baseado em busca

O projeto de sequências baseado em busca, em geral, define a função F_{seq} através do seguinte pipeline:

1. Dada uma estrutura proteíca, é definida uma sequência de aminoácidos inicial, baseada em uma heurística H.

2. A sequência serve de entrada para a função objetivo F_{obj} que calcula a métrica a ser otimizada pelo pipeline.

3. Se a sequência atual é melhor do que sequência final, em termos da métrica calculada pela F_{obj}, então o Agente A define a sequência atual como a nova sequência final. Além disso, A também atualiza a sequência atual a partir de mutações.

4. O processo se repete a partir do item 2 até que seja atingido um critério de parada. Geralmente é baseado no número de iterações e/ou na métrica objetivo.

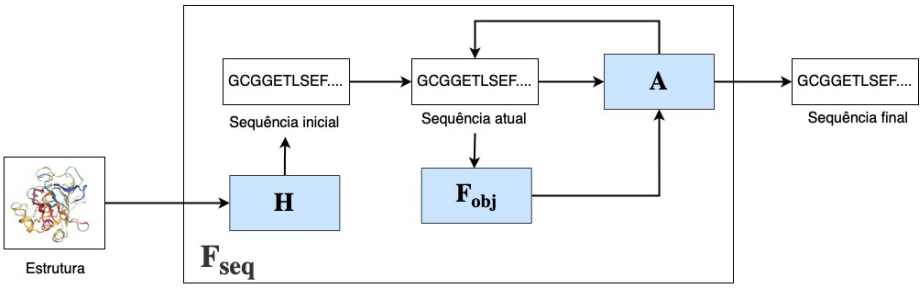


Figura 3: *Projeto de Sequências baseado em busca*

O *Rosetta* B, 2019 se baseia em um pipeline como este, onde a heurística H define como partida uma sequência de uma proteína homóloga ou estruturalmente semelhante a estrutura alvo. Sua função objetivo F_{obj} mede a energia livre da proteína, considerando diversos fatores como interações de *Van der Waals*, interações eletrostáticas, ligações de hidrogênio, solubilidade, entre outros. O *Rosetta* busca minimizar a energia livre calculada pela F_{obj}. Portanto, seu Agente A decide que a sequência final passa a ser a atual se houver um decréscimo na energia livre. Além disso, atribui modificações na sequência atual de forma aleatória, utilizando o Método de Monte Carlo (MMC).

Existem abordagens cujo o objetivo da busca se concentra em maximizar a similaridade entre as proteínas, como adotado por xx. Com o objetivo de criar novas proteínas nunca antes vistas na natureza, Anishchenko, 2021 parte de uma sequência aleatória de aminoácidos, i.e, não possui uma heurística inicial. A função M, assim como o *Rosetta*, consiste de um MCC. Já a função objetivo F_{obj} calcula a divergência de *Kullback-Leibler* entre ...

0.1.2 Baseado em aprendizado profundo

O projeto de sequências baseado em aprendizado profundo define a função F_{seq} a partir de redes neurais artificiais. O J. D. e. al, 2022 por exemplo, determina a F_{seq} como uma rede neural profunda, denominada *ProteinMPNN*, que mapeia de forma direta a estrutura alvo à sequência de aminoácidos. A rede é construída através de uma *Message Passing Neural Network* (MPNN) composta por uma arquitetura *encoder-decoder* que se baseia nas características da estrutura - distância e orientação dos átomos no espaço - para fazer predições J. D. e. al, 2022.

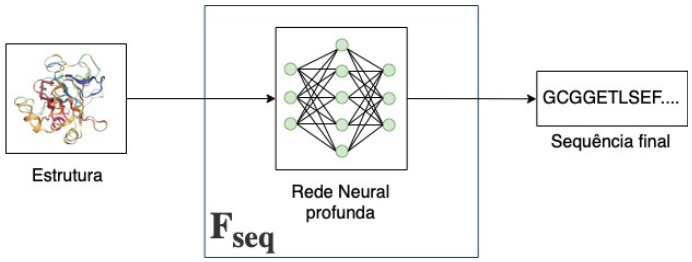


Figura 4: *Projeto de Sequências baseado em aprendizado profundo*

Falar sobre o RL in latent space e mencionar alg. genetico

0.2 Proposta

Além de possívelmente provocar uma resposta imune no paciente, o tratamento da hemofilia tipo B, baseado na infusão do Fator IX, é custoso devido ao pouco tempo de vida do Fator IX na corrente sanguínea, demandando infusões frequentes. Neste sentido, o presente trabalho se propõe a responder a seguinte pergunta científica: É possível, através de projeto de sequências, projetar uma proteína que desempenhe a mesma função do Fator IX mas com um perfil imunológico melhor e que demande menos infusões no tratamento da hemofilia tipo B?

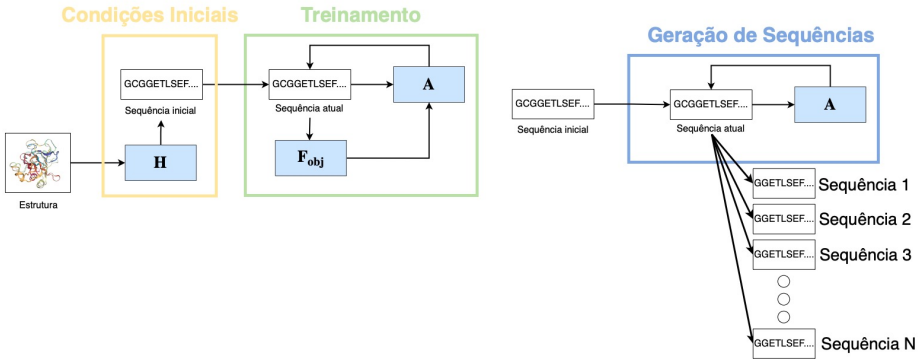
Para responder a esta questão, formulamos a hipótese de que, se existir uma proteína alternativa superior ao Fator IX, ela deverá ser estruturalmente similar. Com base nisso, pretendemos selecionar, dentre um conjunto de proteínas estruturalmente semelhantes ao Fator IX, aquelas que satisfaçam os critérios estabelecidos, caso existam.

De modo a obter esse conjunto de proteínas, será desenvolvido um pipeline que combina a arquitetura do projeto de sequências baseado em busca e técnicas de aprendizado profundo. O processo será dividido em três módulos: Condições Iniciais, Treinamento e Geração de Sequências.

O módulo de Condições iniciais consiste na obtenção da sequência inicial de aminoácidos através da heurística H, bem como o cálculo do erro inicial associado a esta sequência, usando a função objetivo F_{obj}. Vamos utilizar o *ProteinMPNN* como heurística H e a métrica de similaridade estrutural *Template Matching Score* (*TMScore*) como função objetivo.

Diferente do uso de agentes que realizam mutações aleatórias, como no *Rosetta*, no módulo de Treinamento, treinaremos uma rede neural profunda para atuar como o Agente *A*, capaz de realizar mutações que otimizem a similaridade com a estrutura alvo, isto é, o Fator IX.

Após o treinamento, no módulo de Geração de Sequências, espera-se que o Agente seja capaz de gerar várias proteínas estruturalmente semelhantes à estrutura alvo por meio de mutações direcionadas. A partir das sequências geradas, avaliaremos a existência de candidatas viáveis para substituir o Fator IX no tratamento da hemofilia tipo B.



- o *ProteinMPNN* será utilizado como heurística H.
- Na função objetivo F_{obj} será calculada uma métrica de similaridade entre a estrutura da proteína atual e a estrutura alvo que pretendemos maximizar: *Template Matching Score - TMScore*.
- O Agente A será uma rede neural profunda, nomeada de *GenSeq*, treinada para receber uma sequência de aminoácidos e prever a mutação que maximiza a similaridade com a estrutura alvo, i.e, otimiza F_{obj}. O treinamento do A será feito com aprendizado por reforço profundo, utilizando a ténica *Proximal Policy Optimization* - PPO
- Depois de treinado, A será utilizado como gerador de proteínas estruturalmente semelhantes ao Fator IX.

Figura 5: *Projeto de Sequências proposto*

0.3 Objetivos

Definimos os seguintes objetivos:

1. Obter um conjunto de sequências de aminoácidos que mimetizem a estrutura do fator de coagulação IX baseado na métrica TM-Score.
2. Avaliar, dentro do conjunto de proteínas obtidas em 1, quais possuem potencial de substituir o Fator IX de modo tornar o tratamento de Hemofilia tipo B mais acessível.
3. Desenvolver um *pipeline* genérico que produza sequências que mimetizem uma estrutura qualquer.

Capítulo 1

Revisão Bibliográfica

TO DO Lopes Tiago J. S., 2021

1.0.1 Predição de estruturas

TO DO

Capítulo 2

Fundamentação teórica

Em andamento

Neste capítulo vamos apresentar alguns fundamentos teóricos utilizados neste trabalho acerca de proteínas e aprendizado por reforço profundo.

2.1 Proteina

Uma proteína é uma macromolécula biológica composta por cadeias de aminoácidos. Constituem a maior parte da massa seca de uma célula, podendo desempenhar funções enzimáticas, estruturais, imonológicas, de transporte, entre outras B. Alberts, 2002.

Existem milhares de proteínas diferentes em uma célula, onde cada uma é composta por uma sequência distinta dos 20 tipos de aminoácidos encontrados na natureza, unidos através de ligações peptídicas. Por conta disto, as proteínas são também conhecidas como polipeptídeos B. Alberts, 2002.

A cadeia polipeptídica é composta por três componentes: a cadeia principal, as cadeias laterais e as ligações peptídicas. A cadeia principal é também referida como o esqueleto da proteína e é definida como uma série repetitiva e encadeada de átomos de carbono, nitrogénio e oxigénio. Anexadas a cadeia principal, encontram-se as cadeias laterais. Estas não estão envolvidas nas ligações peptídicas e são responsáveis por caracterizar as principais propriedades da proteína B. Alberts, 2002.

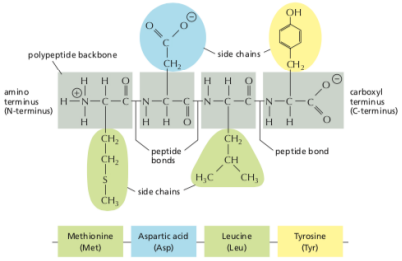


Figura 2.1: *Componentes de uma proteína* B. Alberts, 2002

AMINO ACID			SIDE CHAIN
Aspartic acid	Asp	D	negative
Glutamic acid	Glu	E	negative
Arginine	Arg	R	positive
Lysine	Lys	K	positive
Histidine	His	H	positive
Asparagine	Asn	N	uncharged polar
Glutamine	Gln	Q	uncharged polar
Serine	Ser	S	uncharged polar
Threonine	Thr	T	uncharged polar
Tyrosine	Tyr	Y	uncharged polar

POLAR AMINO ACIDS

AMINO ACID			SIDE CHAIN
Alanine	Ala	A	nonpolar
Glycine	Gly	G	nonpolar
Valine	Val	V	nonpolar
Leucine	Leu	L	nonpolar
Isoleucine	Ile	I	nonpolar
Proline	Pro	P	nonpolar
Phenylalanine	Phe	F	nonpolar
Methionine	Met	M	nonpolar
Tryptophan	Trp	W	nonpolar
Cysteine	Cys	C	nonpolar

NONPOLAR AMINO ACIDS

Figure 3–2 **The 20 amino acids commonly found in proteins.** Each amino acid has a three-letter and a one-letter abbreviation. There are equal numbers of polar and nonpolar side chains; however, some side chains listed here as polar are large enough to have some nonpolar properties (for example, Tyr, Thr, Arg, Lys). For atomic structures, see Panel 3–1 (pp. 112–113).

Figura 2.2: *Os 20 aminoácidos* B. Alberts, 2002

2.1.1 Estrutura proteíca

A função de uma proteína está intimamente relacionada à sua estrutura tridimensional, que é formada principalmente a partir das interações entre as cadeias laterais e móleculas encontradas no meio físico ao qual a proteína está inserida. Devido a essas interações, a maioria das proteínas assume uma estrutura tridimensional única, que é determinada pela ordem dos aminoácidos na sequência. A conformação final tende a ser aquela que possui a menor energia livre B. Alberts, 2002.

B. Alberts, 2002 explica que a polaridade das cadeias laterais desempenham um papel importante na predição estrutural da proteína. As cadeias não polares (hidrofóbicas) tendem a se agrupar no interior da molécula, evitando contato com a água. Já as polares tendem a se dispor perto da superfície da molécula, onde podem formar ligações de hidrogênio com a água e outras moléculas polares.

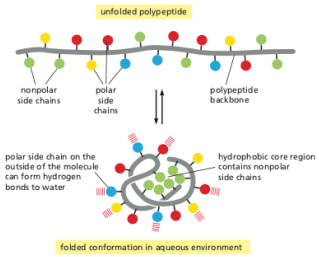


Figura 2.3: *Exemplo de conformação protéica B. Alberts, 2002*

2.1.2 Resposta Imunológica

TO DO

2.1.3 Afinidade de ligação

TO DO

2.1.4 Estrutura alvo - Fator IX

TO DO

2.1.5 Mutação

Uma mutação no contexto de proteínas refere-se a uma modificação na sequência de aminoácidos. Por menor que seja esta modificação, a mutação pode acarretar em uma alteração na estrutura tridimensional da proteína, uma vez que a ordem que os aminoácidos estão dispostos na sequência influencia diretamente na conformação da estrutura. Neste trabalho, vamos assumir que uma mutação consiste na substituição de apenas um aminoácido por outro na sequência.

2.1.6 Conservation Score

O *Conservation Score* (CS) é uma métrica que estima a importância ou contribuição individual de cada aminoácido da sequência na caracterização da função da proteína. Aminoácidos cruciais na estrutura da proteína tendem a ser conservados, visto que alterações neles podem impactar significativamente a função proteíca. O score é obtido levando-se em conta a frequência com que certas combinações de aminoácidos ocorrem em proteínas homólogas ao longo da evolução das espécies Eddy, 2004. Para este trabalho, utilizamos o CS calculado pelo servidor ConsurfDB Ben Chorin, 2020, tendo a estrutura do fator IX de coagulação como entrada. colocar uma imagem pra explicar

2.1.7 Similariedade entre aminoácidos

Para mensurar o nível de similariedade entre os aminoácidos, utilizamos uma métrica de distância com base no trabalho Lopes Tiago J. S., 2021. Nesse estudo, uma representação vetorial de 544 dimensões foi criada para cada aminoácido, incorporando informações sobre suas propriedades. Essa representação foi desenvolvida com o auxílio do pacote seqinR Charif, 2007. Posteriormente, Lopes Tiago J. S., 2021 aplicou uma redução de dimensionalidade com Principal Component Analysis (PCA), obtendo uma nova representação com 19 dimensões, retendo 99% da informação original. Por fim, Lopes Tiago J. S., 2021 apresenta a matriz de distâncias que iremos utilizar neste trabalho, que é o resultado da distancia euclidiana entre cada par de vetores de 19 dimensões.

	A	C	D	E	F	G	H	I	K	L	M	N	P	Q	R	S	T	V	W	Y
A	0	1.5	1.57	1.51	1.52	1.59	1.51	1.52	1.52	1.46	1.48	1.51	1.49	1.47	1.6	1.38	1.49	1.44	1.55	1.59
C	1.5	0	1.6	1.54	1.48	1.68	1.51	1.46	1.61	1.49	1.48	1.59	1.34	1.5	1.61	1.53	1.54	1.42	1.57	1.56
D	1.57	1.6	0	1.48	1.65	1.61	1.55	1.62	1.59	1.67	1.65	1.51	1.46	1.53	1.63	1.52	1.58	1.6	1.72	1.62
E	1.51	1.54	1.48	0	1.62	1.63	1.53	1.6	1.58	1.63	1.58	1.5	1.51	1.47	1.61	1.56	1.58	1.57	1.7	1.63
F	1.52	1.48	1.65	1.62	0	1.63	1.48	1.39	1.61	1.45	1.48	1.56	1.6	1.54	1.61	1.58	1.52	1.42	1.53	1.49
G	1.59	1.68	1.61	1.63	1.63	0	1.58	1.6	1.57	1.65	1.68	1.53	1.47	1.57	1.59	1.54	1.62	1.57	1.77	1.59
H	1.51	1.51	1.55	1.53	1.48	1.58	0	1.49	1.51	1.52	1.46	1.48	1.44	1.46	1.52	1.51	1.54	1.47	1.62	1.5
I	1.52	1.46	1.62	1.6	1.39	1.6	1.49	0	1.58	1.42	1.42	1.5	1.58	1.51	1.57	1.57	1.53	1.14	1.55	1.45
K	1.52	1.61	1.59	1.58	1.61	1.57	1.51	1.58	0	1.59	1.61	1.48	1.54	1.45	1.49	1.44	1.49	1.55	1.65	1.56
L	1.46	1.49	1.67	1.63	1.45	1.65	1.52	1.42	1.59	0	1.5	1.58	1.64	1.56	1.62	1.47	1.52	1.43	1.53	1.54
M	1.48	1.48	1.65	1.58	1.48	1.68	1.46	1.42	1.61	1.5	0	1.62	1.55	1.52	1.65	1.6	1.6	1.48	1.57	1.57
N	1.51	1.59	1.51	1.5	1.56	1.53	1.48	1.5	1.48	1.58	1.62	0	1.4	1.45	1.52	1.35	1.49	1.56	1.62	1.54
P	1.49	1.34	1.46	1.51	1.6	1.47	1.44	1.58	1.54	1.64	1.55	1.4	0	1.45	1.57	1.38	1.46	1.55	1.65	1.58
Q	1.47	1.5	1.53	1.47	1.54	1.57	1.46	1.51	1.45	1.56	1.52	1.45	1.45	0	1.52	1.43	1.44	1.49	1.59	1.53
R	1.6	1.61	1.63	1.61	1.61	1.59	1.52	1.57	1.49	1.62	1.65	1.52	1.57	1.52	0	1.53	1.56	1.51	1.73	1.55
S	1.38	1.53	1.52	1.56	1.58	1.54	1.51	1.57	1.44	1.47	1.6	1.35	1.38	1.43	1.53	0	1.21	1.49	1.55	1.5
T	1.49	1.54	1.58	1.58	1.52	1.62	1.54	1.53	1.49	1.52	1.6	1.49	1.46	1.44	1.56	1.21	0	1.43	1.53	1.56
V	1.44	1.42	1.6	1.57	1.42	1.57	1.47	1.14	1.55	1.43	1.48	1.56	1.55	1.49	1.51	1.49	1.43	0	1.59	1.4
W	1.55	1.57	1.72	1.7	1.53	1.77	1.62	1.55	1.65	1.53	1.57	1.62	1.65	1.59	1.73	1.55	1.53	1.59	0	1.59
Y	1.59	1.56	1.62	1.63	1.49	1.59	1.5	1.45	1.56	1.54	1.57	1.54	1.58	1.53	1.55	1.5	1.56	1.4	1.59	0

Figura 2.4: *Distâncias entre pares de aminoácidos Lopes Tiago J. S., 2021*

2.1.8 Similariedade entre estruturas

A métrica utilizada neste trabalho para mensurar a similariedade entre estruturas é o *Template Matching Score - TMScore*. Baseada principalmente em distâncias entre aminoácidos, o *TMScore* é uma métrica que varia de 0 a 1, sendo que quanto mais próximo de 1, maior será a similaridade entre as estruturas. Um TM-Score igual a 1 indica uma sobreposição perfeita e uma correspondência estrutural quase idêntica entre as proteínas, enquanto valores menores sugerem uma maior divergência estrutural entre elas. O cálculo desta medida foi feito a partir da plataforma desenvolvida por Zhang, 2022.

2.2 Aprendizado por reforço profundo

Nesta seção vamos introduzir os conceitos envolvidos no aprendizado por reforço profundo, que formam a base teórica da técnica que iremos utilizar: Proximal Policy Optimization (PPO). Para tal, será fornecido um contexto sobre Processo de Decisão de Markov (MDP), Redes Neurais (NN) e o método Ator-Crítico (AC).

2.2.1 Processo de Decisão de Markov

O Processo de Decisão de Markov (MDP) é utilizado para modelar processos de forma probabilística. Ele é chamado de markoviano pois a distribuição de probabilidade de um estado depende apenas do estado anterior e da ação selecionada. Uma ação dá origem a um novo estado ao passo que promove alterações no estado atual. Um Agente seleciona cada ação com base em uma política (π) que mapeia estados a ações. Em um MDP, o processo evolui em etapas, ou *steps*, à medida que o agente toma ações. A sequência de estados, desde o inicial até o final, é chamada de episódio. A qualidade de cada ação é avaliada com base no quão benéfico é o estado alcançado. De acordo com Pellegrini, 2007, o MDP pode ser definido como uma tupla <S,A,T,R> onde:

- S é o conjunto de possíveis estados;
- A é o conjunto de ações que interferem no processo;
- T :S×A×S→[0,1] é uma função que quantifica a probabilidade de mudança do estado s ∈ S para o estado s' ∈ S, dado a ação a ∈ A selecionada. É representado por T (s' |s, a);
- r : S × A→r é uma função que mede a recompensa por selecionar a ação a ∈ A quando o processo está no estado s ∈ S.

De modo a encontrar um bom π, é necessário definir uma maneira de comparar diferentes políticas. Uma técnica comum é comparar a esperança da recompensa acumulada descontada, ou Valor de Retorno (R), que cada política produziu em um mesmo episódio.

$$R = E\left[\sum_{k=1}^{z} \gamma^{k-1} r_k\right] \tag{2.1}$$

Sendo z o tamanho do episódio, k o índice do *step* e o r_k a recompensa obtida em k. A constante γ é um valor entre 0 e 1, chamado de fator de desconto. Ele é responsável por atribuir um peso maior a recompensas obtidas no início do episódio.

Além disso, existem outras duas métricas importantes para se avaliar a qualidade de uma política: a Função de Valor (V) e a função de Valor da Ação (Q). A primeira estima a vantagem de estar em um estado s, calculando a esperança de R dado o estado inicial s0 = s.

$$V_\pi(s) = E[R|s_0 = s] \tag{2.2}$$

A segunda estima a vantagem de selecionar a ação a estando no estado s. Em outras palavras, é definida como a esperança de R dado o estado inicial e a ação inicial:

$$Q_\pi(s, a) = E[R|s_0 = s, a_0 = a] \tag{2.3}$$

Neste trabalho vamos modelar tanto a política π quanto a Função de Valor V utilizando redes neurais profundas.

2.2.2 Redes Neurais

Inspirado no sistema neural humano, uma rede neural artificial consiste em um conjunto de nós, chamados de neurônios, interconectados por arestas Bishop, 2006. Em geral, é organizada em camadas, que incluem três tipos: a camada de entrada, a camada oculta e a camada de saída, conforme ilustrado na figura 2.5.

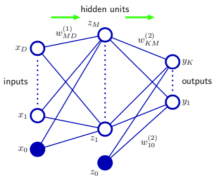


Figura 2.5: *Arquitetura de uma rede neural Bishop, 2006*

Sinais fluem entre os neurônios através das arestas que estão associadas a um peso (w) que quantifica a sua importância. Nos neurônios é feito o processamento dos diversos sinais recebidos que, em seguida, aplica-se uma função, conhecida como função de ativação. Esta pode ser simplesmente uma combinação linear das entradas ponderadas pelos seus respectivos pesos, como pode ser função não linear como sigmoid, tangente hiperbólica, entre outras Bishop, 2006.

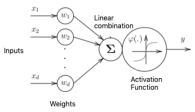


Figura 2.6: *Neurônio processando sinal Figueiredo, 2018*

Os pesos das arestas são ajustados durante a fase de treinamento, de modo a orientar a rede a produzir a saída esperada para uma entrada específica. Nesta fase, é aplicado

uma função de perda baseado na saída da rede. Existem diversas técnicas possíveis para minimizar a função de perda. Dentre elas, uma das mais tradicionais é a baseada no gradiente descendente, que atualiza os pesos conforme a equação a seguir.

$$w(t+1) = w(t) - n\nabla L(w(t)) \tag{2.4}$$

Onde w é o vetor de pesos, n é uma constante positiva conhecida como taxa de aprendizagem, que quantifica o tamanho do ajuste dos parâmetros a cada iteração e L é a função de perda. Um dos algoritmo mais utilizados para calcular o gradiente da função de perda e atualizar os parâmetros da rede é o *Backpropagation* Bishop, 2006.

No contexto deste trabalho, as funções de perda estarão fortemente relacionadas ao valor de Retorno R e a função de Valor V apresentadas na seção anterior.

2.2.3 Método Actor-Critic

O método Ator-Crítico (AC) é um procedimento que tem como objetivo otimizar a política de um MDP Konda e Tsitsiklis, 2001. Ele é composto por dois componentes principais: o Ator e o Crítico. O Ator é aquele que seleciona as ações baseado nos estados, i.e, desempenha a função da política em um MDP. O Crítico, como o nome sugere, avalia a ação selecionada pelo Ator, estimando o vantagem de estar no estado alcançado. Assim, o Crítico é representado pela função de Valor do MDP. Tanto o Ator quanto o Crítico serão estimados por uma rede neural profunda cujos parâmetros serão ajustados utilizando a técnica *Proximal Policy Optimization* J. S. e. al, 2017.

2.2.4 PPO

A técnica *Proximal Policy Optimization* é um modelo de aprendizado por reforço profundo que define funções de perda para ajustar os pesos das redes neurais pertencentes à arquitetura do método AC J. S. e. al, 2017.

Uma vez que o Crítico objetiva estimar o valor de V, o ajuste dos seus parâmetros (W) é feito através de um gradiente descendente a fim de minimizar o quadrado da diferença entre a saída da rede ($\hat{V}(W)$) e o V observado (V^{obs}) J. S. e. al, 2017:

$$L_t^{VF}(W) = (\hat{V}_t(W) - {V_t}^{obs})^2 \tag{2.5}$$

Uma dos principais motivações do PPO é, durante o processo de otimização do MDP, evitar atualizações de política muito grandes. Para isto, J. S. e. al, 2017 força com que a razão entre a política atual e a política anterior ($r_t(\Theta)$) seja limitada a um intervalo conveniente.

$$r_t(\Theta) = \frac{\pi_\Theta}{\pi_{\Theta old}} \tag{2.6}$$

Desta forma, para ajustar os parâmetros (Θ) do Ator, J. S. e. al, 2017 define a seguinte função objetivo a ser maximizada por um gradiente ascendente:

$$L^{CLIP}(\Theta) = \hat{E}_t[min(r_t(\Theta)\hat{A}_t, clip(r_t(\Theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)] \tag{2.7}$$

Onde ϵ é uma constante que o J. S. e. AL, 2017 sugere ser igual a 0.2. $\hat{A}_t$ é definido como a estimativa da função de vantagem, que corresponde ao benefício de selecionar uma ação a no estado s, em comparação com uma seleção aleatória de uma ação no estado s. Em outras palavras, é a diferença entre Q e V:

$$A = Q(s, a) - V(s) \tag{2.8}$$

A função objetivo $L^{CLIP}(\Theta)$ é a esperança do mínimo de dois termos: $r_t(\Theta)\hat{A}_t$ e $clip(r_t(\Theta), 1-\epsilon, 1+\epsilon)\hat{A}_t$. O primeiro direciona a política a selecionar ações que maximizam a função de vantagem. O segundo é uma versão truncada do primeiro, onde é aplicado um *clip* no $r_t(\Theta)$ de modo a garantir que a nova política não se distancie muito da política anterior.

Uma vez que o aprendizado das redes do AC são dependentes entre si, J. S. e. AL, 2017 sugere uma função objetivo única para treinar ambos:

$$L_t^{PPO}(\Theta) = \hat{E}_t[L_t^{CLIP}(\Theta) - c_1 L_t^{VF}(\Theta) + c_2 S(s_t)] \tag{2.9}$$

Onde c_1 e c_2 são constantes e S é definido por J. S. e. AL, 2017 como um bônus de entropia que garante que o agente explore suficientemente o ambiente durante o treinamento.

2.2.5 MDS

Dado um conjunto de n objetos e uma matriz $D = (dij)$ contendo a distância entre cada par (i, j) de objetos, o objetivo do *Metric Multidimensional Scaling* (MDS) é encontrar, para um objeto qualquer Xi, uma representação vetorial $Xi = [xi1, ..., xim]$ tal que $||Xi - Xj|| = dij$. Para isto, o método busca minimizar a diferença entre $||Xi - Xj||$ e dij através da seguinte função de perda, conhecida por $Stress$:

$$Stress(X1, X2, ..., Xn) = \sqrt{\sum_{i \neq j=1}^{n} (d_{ij} - ||X_i - X_j||)^2} \tag{2.10}$$

Explicar como essa função é minimizada. Daria pra fazer pelo Gradient descendente mas aparentemente o mais usado é o SMACOF

Neste trabalho utilizamos o MDS para codificar cada aminoácido em uma representação vetorial que respeita a matriz de distâncias pré definidas por Lopes Tiago J. S., 2021.

Sklearn cita Nonmetric multidimensional scaling: a numerical method" Kruskal, J. Psychometrika, 29 (1964)

Capítulo 3

Metodologia

Neste capítulo será detalhado os três módulos propostos na figura 5: Condições Iniciais, Treinamento e Geração de Sequências.

3.1 Condições iniciais

O objetivo do primeiro módulo (figura 3.1) é obter a sequência e o erro inicial. A primeira é obtida ao se passar a estrutura alvo como entrada do *ProteinMPNN*. Já o erro inicial é definido pela seguinte expressão:

$$Er_0 = 1 - TMScore(E_{FIX}, E_{ini}) \tag{3.1}$$

Onde E_{FIX} é a estrutura alvo, i.e, a estrutura do fator IX de coagulação e E_{ini} é a estrutura vinculada a sequência inicial, predita pelo *ESMFold* - descrito em 1.0.1. O *TMScore* consiste em uma medida de similariedade entre estruturas, detalhada em 2.1.8

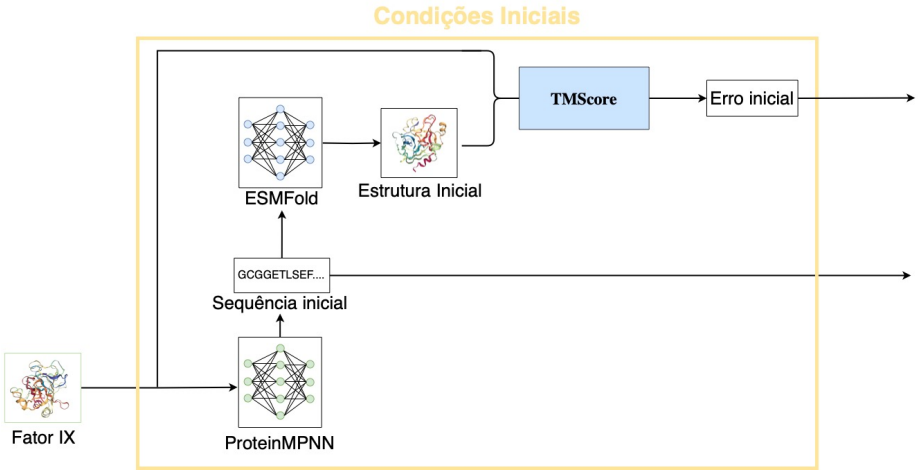


Figura 3.1: *Obtenção das condições iniciais*

3.2 Treinamento

O módulo de Treinamento foi construído em duas etapas: Estágios I e II, detalhados em 3.2.3 e 3.2.4 respectivamente. Para ambos os estágios, o treinamento se dá através de interações entre dois componentes: O Ambiente de aprendizado e o Agente.

3.2.1 Ambiente de aprendizado

O ambiente de aprendizado é modelado como um MDP e, portanto, é responsável por definir o espaço de ações, o espaço de estados, a função de transição entre estados dado uma ação e a função de recompensa.

Espaço de ações

Definimos uma ação como um vetor bi-dimensional composto pelo índice p de uma posição na sequência e um aminoácido r. A ação será utilizada para realizar uma mutação na proteína, inserindo o r na posição p. De modo a incentivar o Agente a explorar diferentes ações, definimos, para cada iteração, um conjunto de posições e aminoácidos válidos: P_{val} e R_{val} respectivamente. P_{val} é composto por todas as possíveis posições na sequência, com exceção das posições previamente selecionadas dentro do mesmo episódio. Definimos R_{val} sendo igual aos 20 possíveis aminoácidos, com exceção dos aminoácidos que foram selecionados mais de 7 vezes no mesmo episódio. Desta forma, o espaço de ações, para cada iteração i, é definido por:

$$A_i = \left\{ (p, r) | p \in P_{val_i}, r \in R_{val_i} \right\} \tag{3.2}$$

Espaço de estados

Definimos o espaço de estados como o conjunto de todas as possíveis sequências de n aminoácidos, onde n é o tamanho da sequência inicial:

$$S = \{(r_1, r_2, r_3, ..., r_n) | r_i \in R_{encoded} \forall i \in [1, n]\} \tag{3.3}$$

Onde $R_{encoded}$ é o conjunto dos 20 possíveis aminoácidos, cada um representado por um vetor codificado pelo *Encoder*. O *Encoder* consiste em um dicionário que mapeia cada aminoácido para uma representação vetorial que conserva as distâncias entre cada par de aminoácidos estabelecidas por Lopes Tiago J. S., 2021. Este mapeamento foi construído através de um MDS utilizando 21 dimensões.

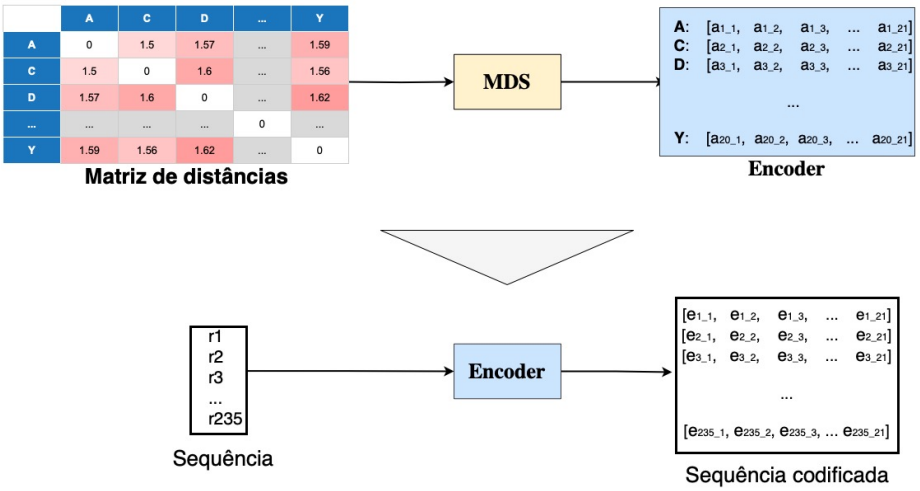


Figura 3.2: *Construção e uso do Encoder.*

Para escolher a quantidade de dimensões a se utilizar no MDS, definimos o ponto a partir do qual a curva entre *Stress* e dimensionalidade tende a se aproximar assintóticamente do eixo x do plano cartesiano. Este ponto, conforme a figura 3.3, é o que possui 21 dimensões.

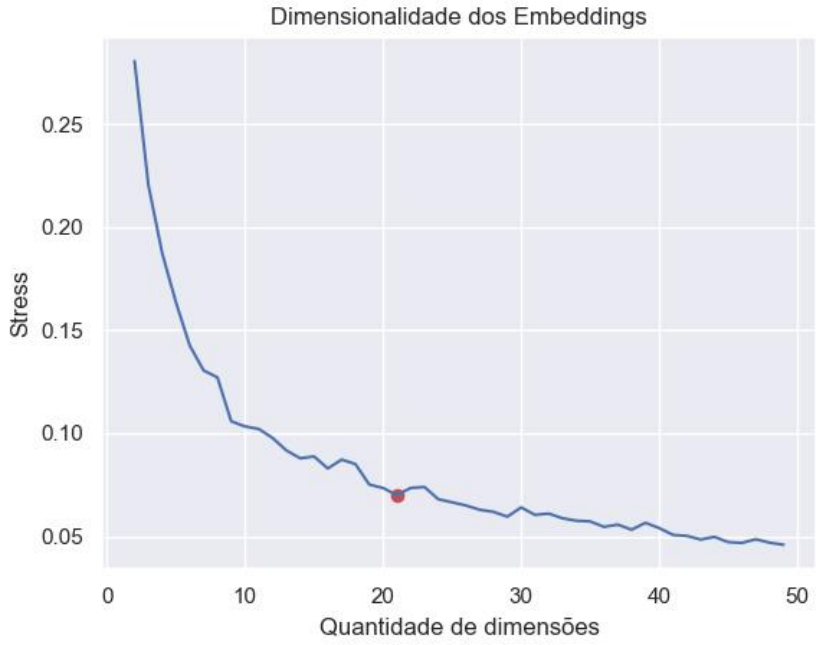


Figura 3.3: *Escolha da dimensionalidade dos Embeddings*

Função de transição de estados

A função de transição de estados é caracterizada pela geração de uma nova sequência, resultante da mutação definida pelo Agente. Em outras palavras, é uma função que tem como entrada uma ação e o estado atual e como saída o estado resultante da transição.

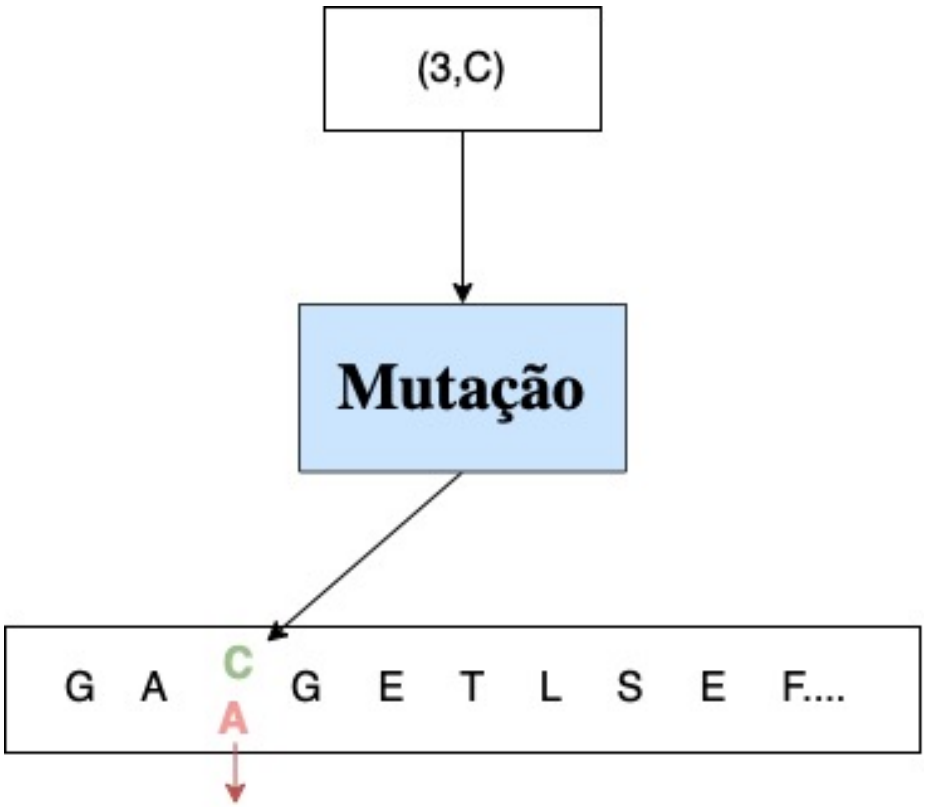


Figura 3.4: *Exemplo de mutação. O aminoácido A na posição 3 está sendo substituído pelo o aminoácido C*

Função de recompensa

Já a função de recompensa calcula a qualidade da ação selecionada pelo Agente. Nas seções 3.2.3 e 3.2.4 especificamos a função de recompensa de cada estágio de treinamento.

3.2.2 Agente

O Agente é modelado por uma rede neural, batizada de *GenSeq*, cujos parâmetros são otimizados através da técnica *Proximal Policy Optimization* (PPO), sendo responsável por decidir qual a melhor mutação a se fazer, dada a sequência de entrada. Em outras palavras, o Agente deve escolher, dentro do espaço de ações, a ação que maximiza a recompensa acumulada. Os hiperparâmetros utilizados no Agente são apresentados na tabela a seguir.

Hiperparâmetro	Valor
Número de camadas escodidas	2
Número de neurônios por camada escondida	64
Função de ativação	tangente hiperbólica
Otimizador	Adam
Taxa de aprendizagem	3e-4
Número de épocas	10
Fator de desconto (γ)	0.99

Tabela 3.1: *Hiperparâmetros do PPO*

Após ser processada pelo *Encoder*, a sequência passa pela camada de entrada do *GenSeq* e segue para duas camadas escondidas de 64 dimensões cada. A camada de saída, por sua vez, possui 255 unidades, organizadas em duas partes:

1. As primeiras 235 unidades da camadas estão associadas ao primeiro atributo da ação: A posição p da sequência de entrada que deve sofrer mutação. A posição predita pelo Agente corresponde ao índice da unidade de saída com maior valor entre as 235 primeiras.

2. Já as 20 últimas unidades de saída estão associadas ao segundo atributo da ação: O índice do aminoácido r que o Agente sugere inserir na posição p. Este valor corresponde ao índice da unidade de saída com maior valor entre as 20 últimas unidades, subtraído de 235.

Por exemplo, suponha que, entre as 235 primeiras unidades, a de maior valor é a terceira, e que, entre as 20 útltimas, o maior valor esteja na unidade 240. Nesse caso, a ação sugerida pelo Agente é fazer uma mutação na posição 3, inserindo o aminoácido de índice 5.

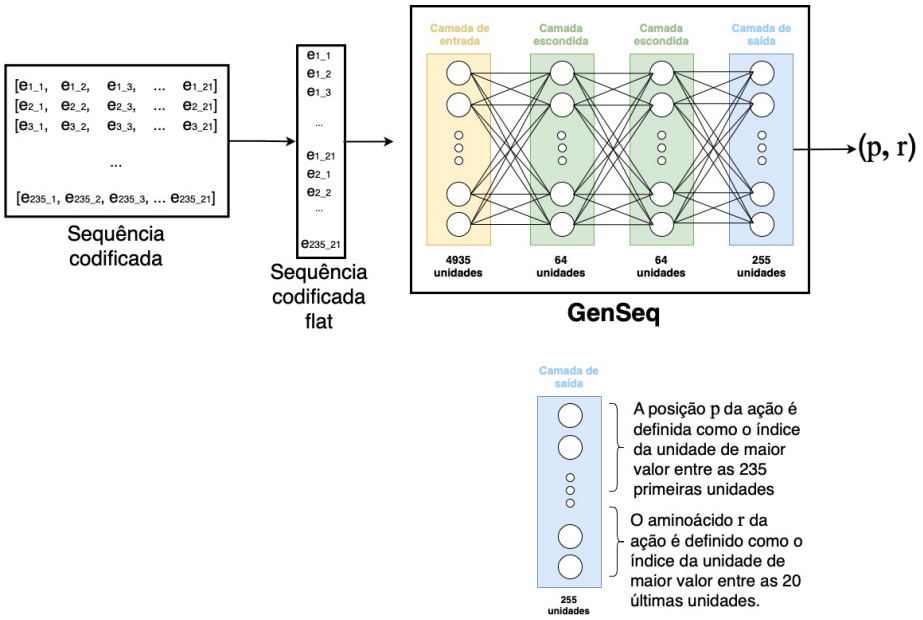


Figura 3.5: *Arquitetura da rede neural do Agente (GenSeq)*

3.2.3 Treinamento - Estágio I

Tendo em vista o vasto número de ações possíveis, desenvolvemos o Estágio I de treinamento com o objetivo de estimular o Agente a encontrar um subespaço menor de ações para atuar. Para isso, atribuímos recompensas negativas (ou penalizações) para ações em posições da sequência cujo o *Conservation Score* é alto, bem como penalizamos ações onde o aminoácido substituído é muito diferente do que fora inserido. Neste sentido, definimos a seguinte função de recompensa:

$$R_I = -(CS(p) + AminoDist(r, r_{prev})) \tag{3.4}$$

Onde p e r são a posição e o aminoácido selecionados pelo Agente, respectivamente. O r_{prev} é o aminoácido que estava na posição p antes da mutação. $CS(p)$ é o *Conservation Score* normalizado da posição p e $AminoDist(r, r_{prev})$ é a medida de similariedade entre os aminoácidos r e r_{prev}.

Neste sentido, o primeiro estágio de treinamento consiste do seguinte procedimento:

1. Os pesos do *GenSeq* são inicializados aleatóriamente.
2. Inicializa o episódio com a sequência inicial e com P_{val} sendo todas as posições da sequência e R_{val} sendo todos os 20 aminoácidos
3. A sequência atual é codificada pelo *Encoder*.
4. *GenSeq* estima a melhor ação (p, r) baseado na sequência codificada.
5. A sequência atual é modificada ao inserir r na posição p.
6. Baseado em (p, r) e r_{prev}, é calculado R_I.
7. p é removido de P_{val} e se a quantidade de vezes que r foi utilizado pelo *GenSeq* é maior ou igual a 7, então r é retirado de R_{val}.
8. Se a iteração atual é igual ao tamanho do *batch*, então *GenSeq* atualiza seus pesos baseado nos valores obtidos de R_I utilizando o *PPO*
9. Se foi atingido o máximo de mutações por episódio, então o episódio é finalziado.
10. Repete a partir do item 2 até atingir o número máximo de iterações.

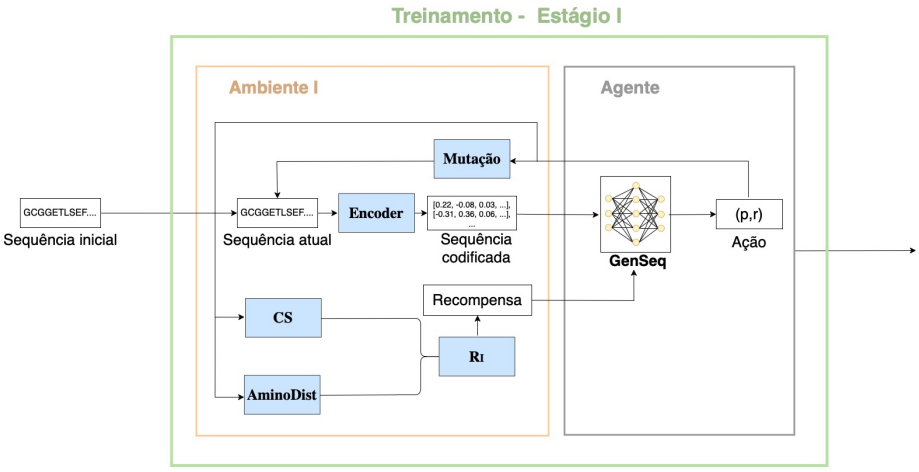


Figura 3.6: *Primeiro estágio de treinamento*

Neste estágio, o Agente foi treinado com um total de 2 milhões de iterações, podendo executar até 100 mutações por episódio.

Parâmetro	Valor
Número máximo de mutações por episódio	100
Número de iterações	2.000.000
Tamanho do *batch*	200

Tabela 3.2: *Parâmetros preliminares*

Após finalizado o primeiro estágio, comparamos o desempenho do Agente com o desempenho de uma política que define ações aleatoriamente, onde observamos que a mediana da distribuição de recompensas do Agente é consideravelmente superior ao do Agente aleatório.

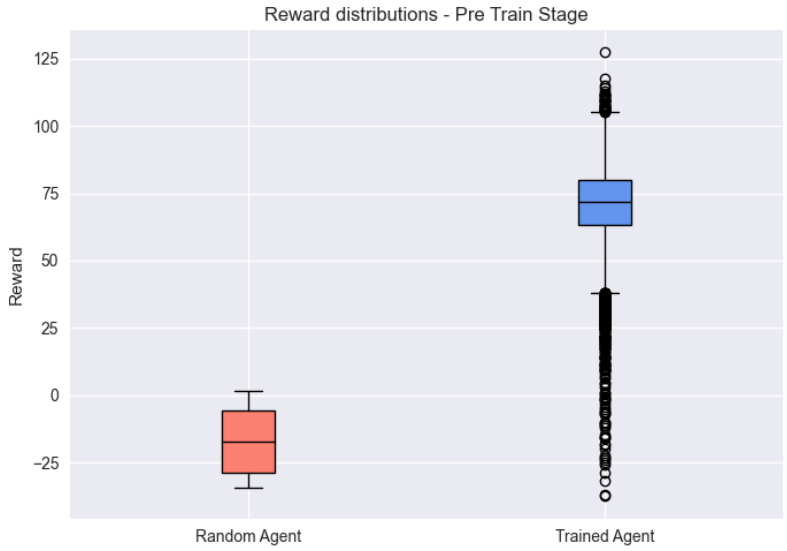


Figura 3.7: *Comparação do Agente pré treinado com um Agente aleatório*

Observa-se que o Agente atingiu um platô de recompensas no ambiente I com cerca de 2500 episódios, como ilustrado em 3.8.

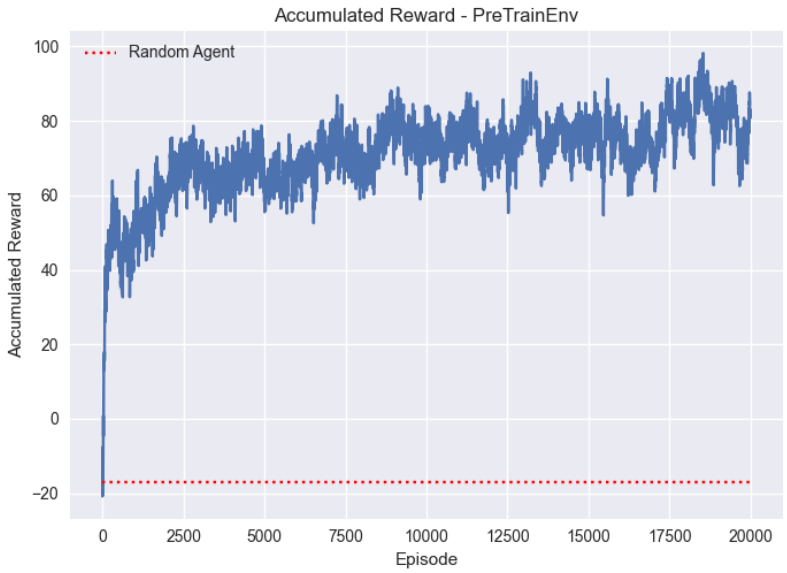


Figura 3.8: *Média móvel das recompensas por episódio - treino*

3.2.4 Treinamento - Estágio II

O objetivo do segundo estágio de treinamento é fazer com que o agente pré treinado aprenda a escolher ações que maximizem o *TM-Score* entre a estrutura gerada e a alvo,

i.e, minimizem o erro $1 - TMScore$. Para isto, definimos a função de recompensa como a soma de 3 termos:

$$R_{II} = -aEr_i + b(Er_0 - Er_i) + c(Er_{i-1} - Er_i) \tag{3.5}$$

Onde Er_i e Er_{i-1} são os erros da iteração i e i-1 respectivamente, e a, b e c são constantes. O primeiro termo, $-aEr_i$, é a penalização pelo erro atual. O segundo determina uma penalização se o erro atual for maior que o inicial e uma recompensa caso contrário. Análogamente, o terceiro termo penaliza se o erro atual for maior que o anterior e recompensa se for menor. Como o objetivo principal é encontrar sequências com erro menores que o inicial, atribuímos um peso maior ao segundo termo, i.e, $b > a$ e $b > c$. Além disso, consideramos $c > a$ para valorizar reduções do erro ao longo do episódio.

Introduzimos o conceito de vitória e derrota a cada episódio. Consideramos vitória caso o erro final seja menor que o inicial e derrota caso contrário. De modo a valorizar a vitória e penalizar a derrota, ao final de cada episódio é adicionado uma recompensa adicional dada por $2500(Er_0 - Er_f)$, onde Er_f é o erro da final, i.e, da última iteração do episódio. Cada episódio é finalizado ao se atingir um número máximo de iterações, ou um limiar de erro. Foi definido um limite superior para o erro de modo a evitar que o Agente atinja erros muito grandes em um mesmo episódio, e um limite inferior para valorizar episódios em que o Agente foi capaz de encontrar "atalhos", i.e, conseguiu reduzir o erro considerávelmente em poucas iterações.

Neste sentido, o segundo estágio de treinamento consiste do seguinte procedimento:

1. Inicializa o episódio com a sequência e o erro inicial; P_{val} sendo todas as posições da sequência; R_{val} sendo todos os 20 aminoácidos.
2. A sequência atual é codificada pelo *Encoder*.
3. *GenSeq* estima a melhor ação (p, r) baseado na sequência codificada.
4. A sequência atual é modificada ao inserir r na posição p.
5. É predito a estrutura da sequência atual utilizando o *ESMFold*
6. Baseado no *TMScore* entre a estrutura do Fator IX e da estrutura atual, no erro inicial e no erro da iteração anterior, é calculado R_{II}.
7. p é removido de P_{val} e, se a quantidade de vezes que r foi utilizado pelo *GenSeq* é maior ou igual a 7, então r é retirado de R_{val}.
8. Se a iteração atual é igual ao tamanho do *batch*, então *GenSeq* atualiza seus pesos baseado nos valores obtidos por R_{II} utilizando o *PPO*
9. Se foi atingido o máximo de mutações por episódio, ou um limiar de erro, então o episódio é finalziado.
10. Repete a partir do item 1 até atingir o número máximo de iterações.

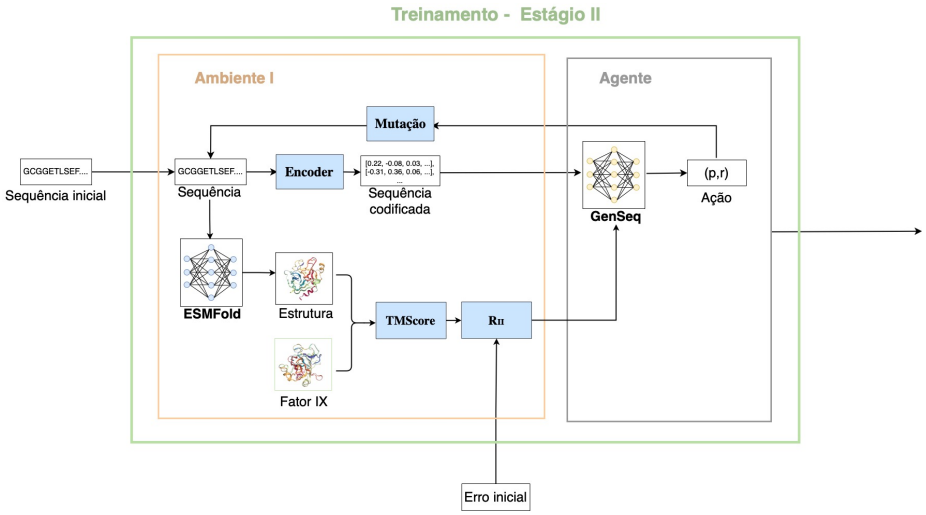


Figura 3.9: *Segundo estágio - Treino*

Neste estágio, o Agente foi treinado com um total de 60 mil de iterações, podendo executar até 45 mutações por episódio.

Parâmetro	Valor
Número máximo de mutações por episódio	45
Erro mínimo no episódio	0.05
Erro máximo no episódio	0.083
Número de iterações	60.000
a	12
b	200
c	50

Tabela 3.3: *Parâmetros preliminares*

Assim como no pré treino, a distribuição das recompensas do Agente ao longo do treinamento é bem superior às recompensas de um Agente aleatório.

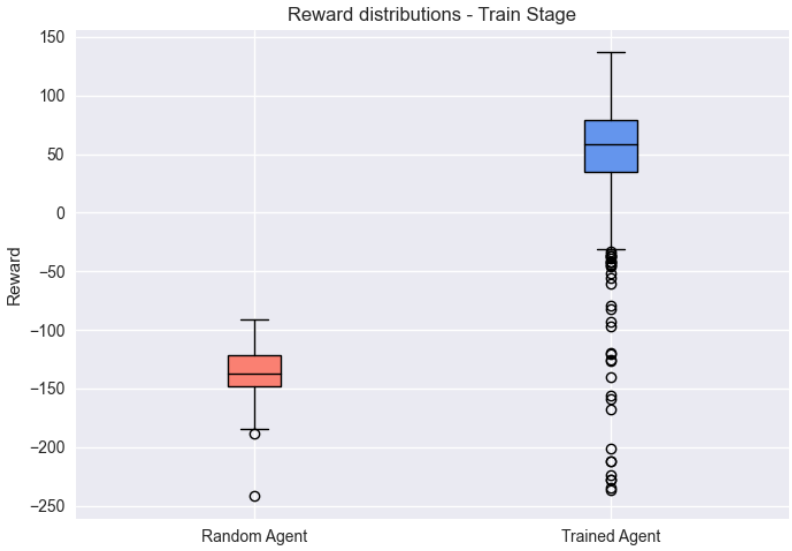


Figura 3.10: *Comparação do Agente treinado com um Agente aleatório*

Repare que, devido ao pré treino, as recompensas médias do Agente são consideravelmente superiores à recompensa média do Agente aleatório desde os primeiros episódios.

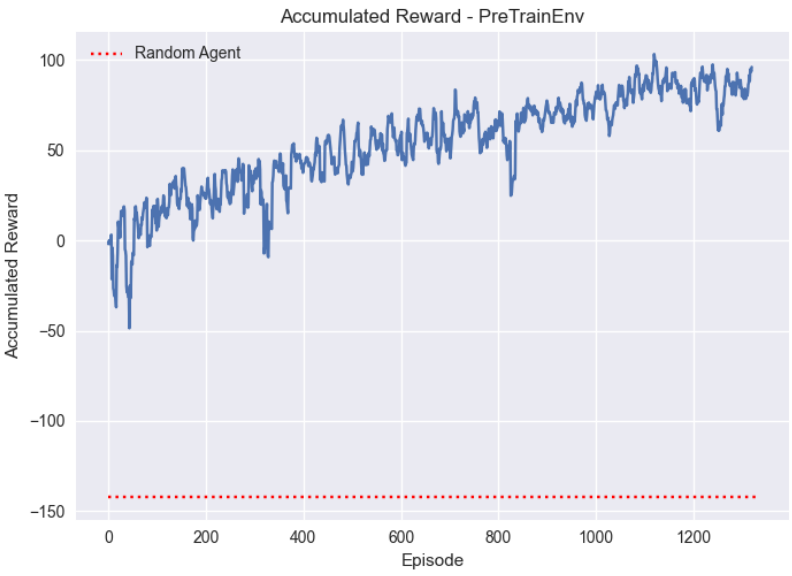


Figura 3.11: *Média móvel das recompensas por episódio - treino*

3.3 Geração de sequências

Por fim, o módulo de Geração de Sequências utiliza o agente treinado para gerar novas sequências que sejam capazes de mimetizar a estrutura alvo melhor do que a sequência inicial, em termos de *TM-Score*.

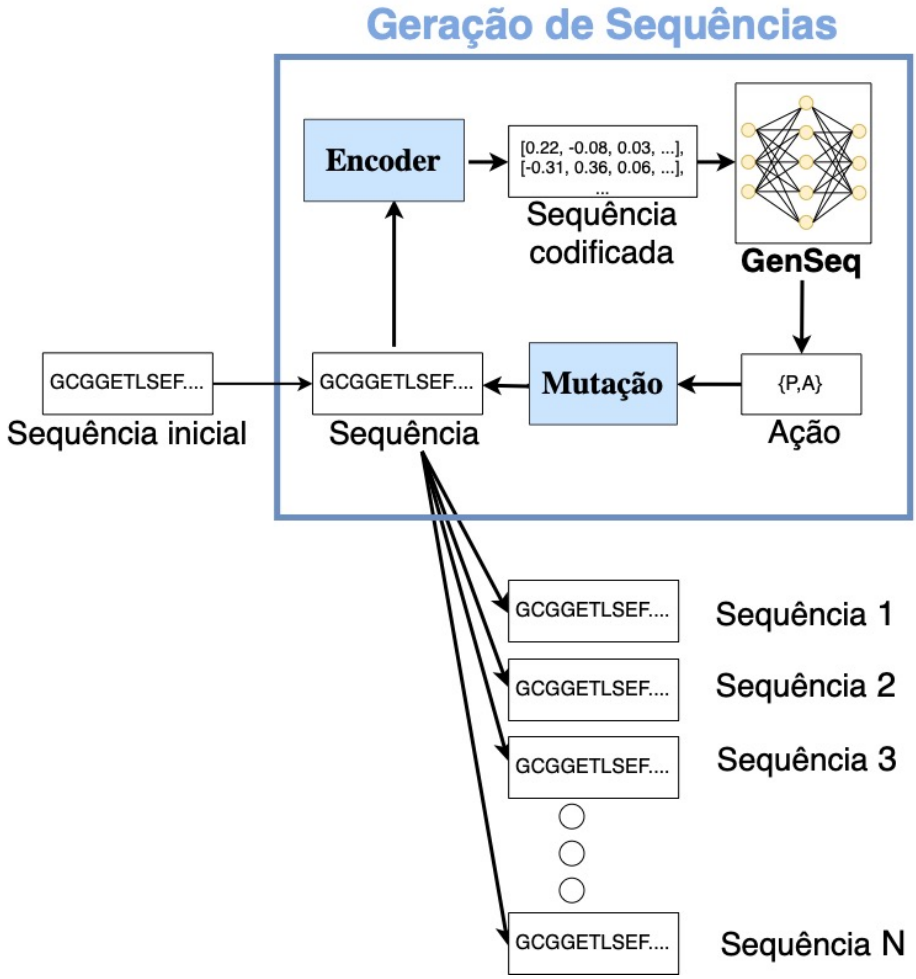


Figura 3.12: *Gerando sequência otimizada*

Capítulo 4

Resultados

Com o Agente treinado, é possível gerar novas sequências seguindo a arquitetura ilustrada no módulo de Geração de Sequências. Neste caso, o Agente gerou mais de 200 proteínas diferentes a partir de mutações na sequência inicial. Destas, 63 possuem um TMScore superior a 92%.

Imagem do poster TMScore x Numero de proteinas

ID	TM-Score	Ganho de TM-Score	IS - seq. inicial	IS - seq. original
1	0.9311	+0.008 pp	91%	31%
2	0.9334	+0.01 pp	84%	29%
3	0.9343	+0.011 pp	82%	29%
4	0.9334	+0.01 pp	81%	28%

Tabela 4.1: *Generated proteins*

(a) *Estrutura Alvo* **(b)** *Estrutura gerada - ID 3*

Figura 4.1: *Comparação entre estruturas*

Mesmo reduzindo a IS (Idêntidade de Sequência) em relação a sequência de partida, o Agente foi capaz de produzir sequências que geram um TM-Score ligeiramente superior ao inicial. Isso nos possibilita utilizar o Agente como um gerador de sequências diferentes que se traduzem em estruturas similares.

Referências

[J. D. e. al 2022] J. Dauparas et al. "Robust deep learning–based protein sequence design using proteinmpnn". *Science* (2022) (citado na pg. 4).

[J. S. e. al 2017] John Schulman et al. "Proximal policy optimization algorithms". *ArXiv* (2017) (citado nas pgs. 15, 16).

[Anishchenko 2021] et al. Anishchenko I. "De novo protein design by deep network hallucination". *Nature* (2021) (citado na pg. 4).

[B 2019] Kuhlman B. "Designing protein structures and complexes with the molecular modeling program rosetta". *J Biol Chem* (2019) (citado na pg. 4).

[B. Alberts 2002] et al. B. Alberts. *Molecular Biology of the Cell*. Garland Science, 2002 (citado nas pgs. 9–11).

[Ben Chorin 2020] A. et al Ben Chorin. "Consurf-db: an accessible repository for the evolutionary conservation patterns of the majority of pdb proteins". *Protein Sci* (2020) (citado na pg. 11).

[Bishop 2006] M. Christopher Bishop. *Pattern Recognition and Machine Learning*. New York: Springer, 2006 (citado nas pgs. 14, 15).

[Charif 2007] J. Charif D. Lobry. "R. in structural approaches to sequence evolution". *Springer* (2007) (citado na pg. 12).

[Eddy 2004] S. R. Eddy. "Where did the blosum62 alignment score matrix come from?" *Nature Biotechnology* (2004) (citado na pg. 11).

[Figueiredo 2018] Jonathas Figueiredo. "Aplicação de algoritmos de aprendizagem por reforço para controle de navios em águas restritas". *Universidade de São Paulo* (2018) (citado na pg. 14).

[Gouw 2013] et al. Gouw S. C. "Factor viii products and inhibitor development in severe hemophilia a". *New England Journal of Medicine* (2013) (citado na pg. 1).

[Konda e Tsitsiklis 2001] Vijay Konda e John Tsitsiklis. "Actor-critic algorithms". *Society for Industrial and Applied Mathematics* 42 (abr. de 2001) (citado na pg. 15).

[Lopes Tiago J. S. 2021] Rios Ricardo et al. Lopes Tiago J. S. Nogueira Tatiane. "Prediction of hemophilia a severity using a small-input machine-learning framework". *Nature* (2021) (citado nas pgs. 7, 12, 16, 18).

[Mancuso 2005] et al. Mancuso M. E. "Environmental risk factors for inhibitor development in children with hemophilia a: a case–control study". *Haematologica* (2005) (citado na pg. 1).

[Pellegrini 2007] Jerônimo Pellegrini. *Processos de Decisão de Markov: um tutorial.* XIV. RITA, 2007 (citado na pg. 13).

[PM 2020] Mannucci PM. "Emophilia therapy: the future has begun." *Haematologica* (2020) (citado na pg. 1).

[Wenze Ding e Gong 2022] Kenta Nakai Wenze Ding e Haipeng Gong. "Protein design via deep learning". *Oxford Academic* (2022) (citado na pg. 2).

[Zhang 2022] et al. Zhang C. "Us-align: universal structure alignments of proteins, nucleic acids, and macromolecular complexes". *Nat Methods* (2022) (citado na pg. 12).