\ subsection {Análise e Resolução de Problemas}

Para desenvolver as métricas de identificação automática dessas habilidades em um indivíduo, utilizamos sua interação com sistema Huxley. Aqui propomos duas métricas, uma está relacionada com a habilidade de Resolução de problemas. A outra, está relacionada com Análise de problemas.

A métrica Resolução de problemas leva em conta a quantidade de soluções, ou seja, quantas submissões um usuário envia ao Huxley. Quanto mais um usuário envia submissões ao sistema, mais ele demonstra gostar de resolver problemas de programação. Por exemplo, dependendo do nível de dificuldade do problema, o usuário pode ter que apresentar diversas submissões ao sistema para conseguir resolvê-lo. No entanto, observe que isso não é necessariamente uma coisa ruim, por conta do processo de aprendizagem e pratica de programação, que é a essência de Huxley. Enviar muitas submissões significa que o usuário está de alguma forma estudando e praticando programação continuamente.

Resolução de problemas: Número total de submissões um usuário enviado para Huxley.

Para fazer com que a pontuação de Resolução de problemas seja avaliada de zero a 100, temos de tomar um valor de referência. Este valor de referência é o número máximo de submissões de um usuário já enviou ao Huxley. Portanto, o usuário que enviou mais soluções para o sistema vai pontuar 100 na habilidade de Resolução de problemas. A pontuação dos demais usuários será calculada relativamente por uma regra de três.

No entanto, quando consideramos apenas o número de submissões isoladamente, um elevado número de submissões erradas também pode significar que o aluno não está analisando muito bem o problema antes de tentar outras vezes. Desta forma, propomos a métrica de Análise de problemas, considerando agora o número de submissões corretas, levando em consideração quantos problemas o usuário foi capaz de resolver e os níveis desses problemas.

Para cada apresentação correta, o usuário do Huxley recebe uma pontuação dependendo do nível do problema que esta submissão responde. Por exemplo, quando um usuário resolve o problema de nível 1, ela ganha um ponto. Da mesma forma, se o problema é o nível 2, ele recebe 2 pontos. A pontuação máxima é de 10, para submissões corretas de problemas nível 10.

Agora, dado que CORscore é a soma da pontuação das submissões corretas. Além disso, dado que SUBscore é a soma da pontuação de todas as submissões. A métrica para de problemas é:

Análise de problemas = CORscore / SUBscore \* 100.

Note que a métrica dessa soft skill pode ser avaliada entre zero a 100. Quanto mais o usuário envia submissões corretas, sua pontuação tende a 100. Caso contrário, se o usuário envia muitas submissões erradas, sua pontuação tende a zero.

\ subsection {Atenção a Detalhes}

Um indivíduo que presta atenção a capaz de utilizar de forma completa todos os recursos disponíveis para realizar suas atividades. No contexto de um juiz online, o usuário atento precisa conhecer bem o sistema e saber como tirar proveito de todas as suas funcionalidades.

Por exemplo, para cada problema do Huxley existe um exemplo de entrada e saída na descrição do problema. Esta entrada e saída é o primeiro caso de teste que avalia cada submissão. Um usuário atento deve sempre observar cuidadosamente essa funcionalidade, para que escrever sua submissão seguindo o exemplo. Com isso, ele evita apresentar soluções que falham no primeiro caso de teste.

Além disso, um usuário atento não apresenta submissões que causam erros de compilação, porque ele conhece os detalhes de sintaxe da linguagem que utiliza, e provavelmente testou seu código antes de submetê-lo.

Dessa forma, para identificar se um usuário possui a soft skill Atenção a detalhes, podemos analisar esses pontos. Para um usuário, seja $SYNTAX\_{ERROR} o número de submissão que foram avaliadas com erros de compilação. Vamos IOERR ser o número de submissão que falhou no primeiro caso de teste. SUBS é o número total de apresentação por parte do utilizador. Definir a métrica como se segue:

Atenção aos detalhes: (1 - SYNERR + IOERR / SUBS) \* 100

% Atenção aos detalhes: (1 - SYNERR + IOERR / SUBS) \* 100

Note-se que, se o usuário sempre fazer apresentações com erros de sintaxe ou cometer erros sobre o exemplo de entrada / saída, o quociente SYNERR + IOERR / SUBS é um deles. Portanto, a atenção aos detalhes marcar torna-se zero. No entanto, se ela sempre evitar esses erros simples, sua atenção aos detalhes é marcar um.

\ subsection {Aprendizagem Rápida}

Para identificar aprendizagem rápida, é necessário estabelecer um critério de comparação, uma vez que esta habilidade macio é a capacidade de aprender comparativamente mais rápido. Em Huxley, os usuários estão agrupados por classes. Para propor uma métrica para identificar aprendizagem rápida, estamos considerando a classe como fator comparativo.

Para calcular a pontuação de aprendizagem rápida, primeiro determinar para cada usuário Huxley sua velocidade para resolver problemas. Tomai a soma da pontuação apresentações corretas para cada dia. Considerando-se dois dias diferentes e seqüenciais (DAY0 e DAYf), em que o usuário enviou, a velocidade é dada por:

VELOCIDADE = Score no DAYf - Score nas DAY0 / Dias entre DAY0 e DAYf

% VELOCIDADE = Score no DAYf - Score nas DAY0 / Dias entre DAY0 e DAYf

% Continuamos cálculo da velocidade em uma janela de 6 meses, que é o período em que o grupo pertence o usuário está tendo aulas. Depois disso, calcula-se a variação de velocidade para obter o ACC, o que vai significar a aceleração média de aprendizagem.

Precisamos tomar um valor de referência para afirmar pontuação rápido aprendizado entre zero a 100. Este valor de referência é o ACC máxima obtida para cada grupo. Em seguida, a pontuação para esta habilidade macio é calculada com a regra de três, tendo como referência a maior aceleração de aprendizagem no grupo o usuário pertence. Assim, o usuário que aprendeu mais rápido em cada grupo vai marcar 100. A pontuação dos demais será calculado relativamente a isso.

\ subsection {} persistencia

Propomos a métrica com base em quantas vezes um usuário continua tentando resolver um problema. Quanto mais um usuário envia apresentações, mais ela demonstra estar praticando continuamente. Mesmo que ela não está batendo uma solução correta, ela está se esforçando e, eventualmente, bater um. Além disso, mesmo depois de uma solução correta, ela pode tentar enviar novamente para melhorá-lo. Esse tipo de comportamento mostra a persistência.

Para marcar na persistência propomos o seguinte resultado: Deixe-COR ser o número de problemas resolvidos por um usuário, excluindo esses problemas que ela corretamente resolver em suas primeiras tentativas. Recomendamos que não contar estes problemas, porque desde que ela tinha razão em sua primeira tentativa, ela não teve chance de aplicar habilidade persistência. Agora vamos Probs ser o número total de problemas experimentados.

Aqui poderíamos dividido COR por Probs para obter a pontuação persistência. No entanto, ainda precisamos ponderar a respeito quando o usuário tentou várias vezes um problema. Mesmo que ela não poderia bater a solução ainda, pelo menos, ela se esforçou. Então, considere que o AVG é a média de tentativas de resolver um problema P em Huxley. Além disso, vamos ICONoveravg ser o número de problemas não resolvidos que o número de tentativas é sobre AVG.

Persistência = COR + ICONoveravg / Probs

% Persistência = COR + ICONoveravg / Probs.

Assim, para alcançar a pontuação máxima na persistência, que é 100, o usuário não deve deixar que problemas ela tentou sem uma solução, ou pelo menos ela tem que tentar duro para resolvê-los. Por outro lado, se um utilizador tentar, pelo menos, uma vez que os problemas a resolver, mas nunca se à solução, a sua pontuação de persistência é zero.

\ subsection {} Comunicação

No contexto da programação, propomos que alguém que tem habilidades de comunicação gera documentação de seus artefatos escrito. Ela é usada para comentar seus códigos-fonte. Além disso, no contexto de um sistema on-line juiz, ela usa recursos de bate-papo para entrar em contato com outros usuários, tem como objetivo fazer amigos na comunidade e participa em fóruns.

Nós poderíamos fazer uso de todas essas características para definir métricas para identificar habilidades de comunicação. No entanto, Huxley não tem um chat ou fórum características. Por isso, estamos usando apenas os comentários no código-fonte. Para um usuário, levar os arquivos de código fonte de seus argumentos corretos. Para cada arquivo de origem selecionada, verifique se ele tem comentários. Seja C a contagem de arquivos de origem que tenham pelo menos um comentário TOTAL ant o número de arquivos analisados. A métrica de comunicação é definido como segue:

Comunicação = C / TOTAL

Comunicação% = C / TOTAL

Note-se que, o valor da comunicação é entre 0 e 100. Se o usuário comentou todos os seus últimos 50 arquivos de origem, assim que sua pontuação Comunicação é 100. Por outro lado, se ela nunca comentar os últimos 50 arquivos de origem, sua pontuação Comunicação é zero.

\ subsection {Trabalho Independente}

Também podemos dizer que se alguém pode realizar tarefas com o mínimo de supervisão e resolução de problemas de programação, sem necessidade de pedir a outra pessoa para ajudar, mesmo diante de altas problemas difíceis, ela pode trabalhar de forma independente.

Para ser capaz de trabalhar de forma independente, uma pessoa precisa para ser confortável em estar sozinho. De acordo com as teorias da personalidade, essa habilidade macio tem uma relação com o fator extroversão [10], que caracterizam as pessoas com escores baixos neste dimensão, porque os introvertidos são reservados e como atividades solitárias.

Huxley oferece duas maneiras de pedir ajuda. No sistema, quando os usuários têm dúvidas eles podem enviar uma pergunta sobre a submissão, que dirigiu a um professor ou alguém que é responsável por seu grupo. Esta outra pessoa lhe responde, escrevendo comentários de ajuda para a questão. A outra maneira é quando um usuário enviar uma resposta errada que eles possam ler uma dica, que é um comentário escrito pelo autor do problema informando qual foi o erro ou como obter a solução certa. Para ler a dica é opcional.

Vamos TIP ser o número de pontas um usuário aceitar a visualizar e ajudar a ser o número de submissão que o usuário pedir e receber comentários de ajuda. Chame SUBS o total de inscrições de um usuário. A métrica de trabalho representa, independentemente:

Trabalhar de forma independente = (1 - TIP + Ajuda / SUBS) \* 100

% Trabalham de forma independente = (1 - TIP + Ajuda / SUBS) \* 100

Note-se que, se um usuário pedir dicas ou ajuda para todas as suas apresentações a TIP quociente + Ajuda / SUBS serão valor como 1, levando Trabalho marcar de forma independente a 0. O oposto disso é quando um usuário nunca pedir ajuda ou dicas. Em seguida, trabalhar de forma independente pontuação é 100.