Dénition expérimentale du cadre de fonctionnement d'un système d'évaluation par grille basé sur l'apprentissage articiel

Damien Follet, Nicolas Delestre, Nicolas Malandain, Laurent Vercouter

Définition expérimentale du cadre de fonctionnement d'un système d'évaluation par grille basé sur l'apprentissage artificiel

Damien Follet, Nicolas Delestre, Nicolas Malandain et Laurent Vercouter

LITIS, Normandie Université & INSA Rouen, Avenue de l'Université
76801 Saint-Étienne-du-Rouvray, France
{damien.follet,nicolas.delestre,
nicolas.malandain,laurent.vercouter}@insa-rouen.fr

Résumé. L'évaluation par grille d'évaluation est de plus en plus répandue. Elle facilite l'apprentissage des apprenants mais augmente la charge d'évaluation des enseignants.

Nous proposons un système utile et utilisable d'aide à l'évaluation par grille, thématiquement indépendant et basé sur l'apprentissage artificiel. Après avoir décrit le système, nous étudions l'utilité et l'utilisabilité de notre système à travers des expériences sur données simulées et réelles. Enfin nous discutons des différentes perspectives d'amélioration de notre système qui découlent de ces observations.

Mots-clé : système de recommandation, grille d'évaluation, indépendance thématique, apprentissage artificiel, cadre expérimental

1 Introduction

De nos jours, les entreprises et le système éducatif utilisent de plus en plus l'évaluation diagnostique. Elle permet aux entreprises d'identifier quelle est la personne la plus adaptée à un poste donné. Elle permet également aux enseignants d'identifier précisément les lacunes d'un apprenant afin de lui conseiller de réviser les points précis du cours dont la maîtrise lui fait défaut.

Les grilles d'évaluation[3] permettent de réaliser une évaluation diagnostique. Une grille d'évaluation définit un ensemble de descripteur qualitatif en langage naturel pour chaque niveau de maîtrise associé à chaque compétence. Le tableau 1 présente un extrait de grille d'évaluation pour une seule compétence.

Tableau 1. Exemple de grille d'évaluation

niveau \ compétence	Identifier les instructions du schéma conditionnel
acquis	Bonne réponse systématique
en cours d'acquisition	Quelques bonnes réponses
non acquis	Aucune bonne réponse

Cependant la tâche d'évaluation s'alourdit puisque le nombre de critères à évaluer augmente significativement. Un système d'aide qui proposerait à l'enseignant une évaluation des copies de la classe pourrait donc alléger son travail et

lui permettre de consacrer plus de temps à aider ses apprenants à combler les lacunes ainsi identifiées. Cela faciliterait donc l'apprentissage de ces apprenants.

L'objectif de cet article est de définir les caractéristiques requises pour un tel système, puis de décrire un système concret répondant à ces caractéristiques.

Tout d'abord ce système d'aide à l'évaluation serait à destination des enseignants, c'est-à-dire à un public non expert en informatique ; les recommandations du domaine de l'Interaction Homme-Machine s'appliquent donc. Ainsi le succès d'un tel système dépend de son utilité, c'est-à-dire répondre à un besoin réel de l'enseignant, et de son utilisabilité, c'est-à-dire dans notre cas répondre à l'enseignant en un temps minimal raisonnable : dans l'idéal en un temps inférieur au seuil humain de perception visuelle, environ $33ms$.

Ensuite plus le système sera thématiquement indépendant, plus il sera utile pour un nombre important d'enseignant. De plus, il pourra traiter des compétences de différents types. Il n'est évidemment pas possible de traiter tous les types de compétences par ordinateur : nous nous limitons aux compétences que l'on peut évaluer par un examen écrit, sur papier ou par ordinateur. Tout le processus d'apprentissage peut être modélisé par 6 catégories qui constituent la taxonomie de Bloom[1,2] : Connaissance, Compréhension, Application, Analyse, Synthèse et Évaluation. Certaines études [10] ont montré que tous ces stades, à l'exception de la Synthèse, peuvent être évalués par des variantes de Questionnaire à Choix Multiple (QCM). Notre objectif est donc un système thématiquement indépendant, en se limitant au traitement de compétences associées à tous les stades excepté Synthèse.

Notre objectif est donc un système utile et utilisable d'aide à l'évaluation de copies constituées de réponses à des questions de type QCM, par grille et thématiquement indépendant. En terme de contraintes techniques, ce système doit pouvoir suggérer l'évaluation correcte de la plupart des copies de la classe[4] en se basant sur le moins possible d'exemples manuels. En effet un système qui aurait besoin qu'on lui fournisse l'évaluation manuelle de la plupart des copies de la classe ne s'avérerait pas très utile, tout comme un système qui ferait beaucoup de mauvaises suggestions à l'enseignant. Il doit également pouvoir répondre en temps raisonnable, ce qui suppose que sa complexité algorithmique soit peu élevée. Afin de faire le moins de suppositions possible sur la thématique, on supposera que les réponses aux questions sont des données nominales, *id est* des données qualitatives entre lesquelles il n'existe pas toujours un ordre intrinsèque.

Nous avons ainsi spécifié les caractéristiques de notre système d'aide. Il nous reste à répondre concrètement à cette spécification ; pour cela nous étudierons tout d'abord les systèmes d'aide à l'évaluation existants afin de déterminer s'ils conviendraient.

2 Systèmes existants

Il existe déjà plusieurs sortes de système d'évaluation ou d'aide à l'évaluation, mais ils sont tous conçus pour réaliser des évaluations quantitatives comme une note finale sur 20, et non pas pour des évaluations qualitatives comme les grilles d'évaluation. De plus ils ne correspondent pas à toutes les contraintes précédemment citées.

Certains sont spécialisés dans une thématique précise, comme STACK[12,13] dans les Mathématiques. Par définition ils ne sont donc pas thématiquement indépendant.

Certains comme Hotpotatoes[14], OpenMark[11] et CUE[6] permettent de spécifier pour chaque question quelles sont les bonnes réponses et quelles sont les mauvaises réponses. Étendu à plusieurs niveaux de compétence, on pourrait spécifier quel est le niveau de compétence à associer à chaque réponse ou combinaison de réponses. Cette approche présente deux inconvénients par rapport à notre objectif. Premièrement le nombre de combinaisons de réponses à spécifier peut très vite exploser avec le nombre de compétences. Spécifier le niveau associé à chaque combinaison de réponses peut alors devenir aliénant pour l'enseignant. Deuxièmement cette spécification est par définition un transfert de connaissances utiles vers l'ordinateur qui suppose la transmission de connaissances explicites et de connaissances tacites[3,8,9]. Ces dernières sont typiquement « des choses que l'on sait mais qu'on trouve impossible ou du moins très difficile à exprimer »[3], car définies en termes très subjectifs et très difficiles à formaliser. L'expression des bonnes et mauvaises réponses est donc très difficile pour l'enseignant, ce qui s'oppose au concept d'utilisabilité que l'on vise.

Certains systèmes utilisent des méthodes d'apprentissage artificiel afin apprendre ce lien entre réponses et niveau de maîtrise au travers d'une interaction tentatives-erreurs avec l'enseignant, lui évitant ainsi d'exprimer directement quelles sont les bonnes et mauvaises réponses et à quel degré. Ces système sont toutefois limités par le type de question ou le type de compétence : par exemple le logiciel SEAR[7] est limité à des productions d'apprenant de type essai et à deux compétences fixes, la qualité du style et celle du contenu. Cette limitation s'oppose donc au concept d'indépendance thématique que l'on vise. Notons cependant que ce système est complémentaire à celui que l'on cherche à concevoir dans le sens qu'il peut permettre de traiter des compétences de type Synthèse.

3 Notre approche

Puisque les systèmes existants ne sont pas adaptés à notre objectif, nous cherchons donc à concevoir nous-même un système utile et utilisable d'aide à l'évaluation thématiquement indépendant et basé sur l'apprentissage artificiel.

Nous commencerons par détailler ce système : le modèle de données que l'on utilisera pour représenter une fonction d'évaluation et l'hypothèse recherchée par l'algorithme d'apprentissage, puis le fonctionnement de notre système. Nous présenterons ensuite les expériences que nous avons conduit afin d'estimer son utilité et son utilisabilité, puis nous terminerons par discuter de certaines limites et perspectives d'amélioration de notre système.

3.1 Modélisation

Afin de pouvoir apprendre la fonction d'évaluation de l'enseignant, on doit définir une modélisation assez précise pour représenter n'importe quelle évaluation que l'enseignant pourrait faire. On doit également définir le modèle de

l'hypothèse que notre algorithme utilisera : s'il est un peu moins précis que l'évaluation, il permettra en échange de pouvoir faire des prédictions sur des situations que l'on n'a pas rencontrées dans l'ensemble d'apprentissage tout en conservant un taux d'erreur faible.

On va donc définir dans ce qui suit la modélisation d'une fonction d'évaluation et la modélisation de sa structure associée, qui sera l'hypothèse que notre algorithme d'apprentissage cherchera à apprendre.

Fonction d'évaluation

On représente la fonction d'évaluation idéalisée par une fonction mathématique associant chaque copie à un niveau de maîtrise pour chaque compétence c.

Une copie représente ici l'ensemble complet des réponses de l'apprenant : pour chaque question de QS (Question Set), l'apprenant a proposé une unique réponse (éventuellement vide s'il n'a rien répondu). Une copie est donc l'ensemble des réponses d'un apprenant.

Le fait qu'une fonction d'évaluation soit une fonction mathématique suppose que celle-ci est déterministe, dans le sens où elle associera toujours la même copie au même niveau de maîtrise pour une compétence. Dans ce cas, les réponses qui composent la copie sont suffisantes à déterminer le niveau de maîtrise d'une compétence : il est ainsi possible d'apprendre séparément chaque compétence lorsque l'enseignant évalue plusieurs compétences à partir de la même copie.

Ainsi à chaque compétence correspondra une fonction d'évaluation. On note $f_{r,c}$ la fonction réelle de l'enseignant pour la compétence c ou simplement f_r lorsqu'il n'y a pas d'ambiguïté sur la compétence visée.

Structure

On dira que la compétence c ne dépend pas de la question q d'après f lorsque quelle que soit la réponse à une question q le niveau de maîtrise prédit par une fonction d'évaluation f reste le même.

On définit alors la structure d'une fonction f par le plus petit ensemble contenant toutes les questions dont dépend la compétence c.

On note $s_{r,c}$ la structure de la fonction réelle de l'enseignant pour la compétence c, ou structure réelle. On note simplement s_r lorsqu'il n'y a pas d'ambiguïté sur la compétence visée. La taille d'une structure est le nombre de questions qui composent cette structure.

Il n'est pas possible de connaître directement la structure réelle s_r. Mais puisque les exemples de l'ensemble d'apprentissage TS (Training Set) ont été générés par la fonction d'évaluation réelle, la structure réelle est par définition consistante avec eux pour la compétence c. Cela restreint donc considérablement notre espace de recherche.

Une structure s est dite consistante avec TS pour la compétence c si et seulement si tout couple de copies tirées de TS dont les réponses aux questions de s sont les mêmes, seront associées au même niveau de compétence pour c.

3.2 Système

Notre système est constitué d'un algorithme d'apprentissage et d'une méthode de sélection de copies qui alimentera notre algorithme. Le tout est organisé de façon à converger vers une fonction d'évaluation pour chaque compétence.

Nous présentons dans cette section le fonctionnement global du système, l'algorithme d'apprentissage puis la méthode de sélection de copie.

Fonctionnement du système

On note GS l'ensemble des copies (pour Global Set), $PreTS$ les copies initiales à faire évaluer par l'enseignant et DS (pour Decision Set) les copies qui restent à évaluer. On note $GS_r = f_r(GS)$ l'ensemble des copies, évaluées par la fonction d'évaluation réelle. Le fonctionnement global de notre système est détaillé dans le pseudo-code suivant.

function SYSTEME(GS, *algorithme*, *selection*, *evaluation manuelle*)
  $(PreTS, DS) \leftarrow selection\ initiale(GS)$
  $TS \leftarrow evaluation\ manuelle(PreTS)$
  repeat
    $DS_p \leftarrow algorithme\_supervisé(TS, DS)$
    if DS_p contient des rejets **then**
      $ex_p \leftarrow selection\ iteration(DS_p)$
      $c_p \leftarrow copie(ex_p)$
      $ex_r \leftarrow evaluation\ manuelle(singleton(c_p))$
      ajouter ex_r à TS
      retirer c_p de DS
    end if
  until (DS_p ne contient pas de rejets $\vee$ DS est vide)
  return DS_p
end function

Informellement, tant qu'il y a des copies que l'on ne sait pas évaluer, on demande à l'enseignant d'évaluer manuellement l'une d'entre elles et on l'ajoute à TS. A la fin, le système est capable d'évaluer (correctement ou non) toutes les copies de la classe.

Plus précisément, ce système repose sur l'utilisation d'un algorithme d'apprentissage artificiel supervisé et d'une méthode de sélection de copies en deux temps : sélection initiale qui sépare l'ensemble des copies en TS et DS disjoints et sélection d'une copie à chaque itération qui sera évaluée manuellement. Dans la pratique, l'évaluation manuelle sera réalisé par l'enseignant par interaction avec le système.

Algorithme d'apprentissage artificiel

Le principe de notre algorithme est d'identifier pour chaque compétence c les structures consistantes avec les exemples de l'ensemble d'apprentissage TS parmi toutes les structures possibles, puis parmi elles la meilleure structure consistante s^* à l'aide d'une heuristique $h(s, TS)$. Une fois identifiée, cette structure permettra de constituer une fonction d'évaluation pour chaque compétence c en

associant pour chaque exemple de TS les réponses aux questions contenues dans s^* avec le niveau de maîtrise n pour c.

Exemple : Soit $TS = \{\{q_1 = a, q_2 = d\} : \{c_1 = 1\}, \{q_1 = b, q_2 = d\} : \{c_1 = 2\}\}$ où $\{q_1 = a, q_2 = d\} : \{c_1 = 1\}$ signifie que la copie constituée de la réponse a à la question q_1 et la réponse d à la question q_2 est évaluée à un niveau 1 pour la compétence c_1. Soit $s = \{q_1\}$ la structure choisie par l'algorithme, alors on peut calculer la fonction d'évaluation partielle $f = \{\{q_1 = a\} : \{c_1 = 1\}, \{q_1 = b\} : \{c_1 = 2\}\}$. Cette fonction partielle permet d'évaluer certaines copies jamais encore observées comme $\{q_1 = a, q_2 = e\} : \{c_1 = 1\}$, d'où le terme d'apprentissage artificiel.

L'heuristique d'une structure consistante s se calcule de la façon suivante : $h(s, TS) = \#s \cdot \#(TS|_s)$ où $\#s$ représente la taille de la structure s et $\#(TS|_s)$ représente le nombre de copies différentes issues de TS lorsque l'on ne conserve que les réponses associées aux questions qui composent s. La meilleure structure est alors celle dont l'heuristique est la moins élevée.

Cependant cet algorithme pose un problème calculatoire : le nombre de structures possibles est exponentiel par rapport au nombre de questions [1].

On peut réduire ce problème grâce à une optimisation. En effet seule une partie des structures consistantes est importante pour nous, et donc seule une partie des structures possibles. Si on examine les structures dans l'ordre croissant de leur taille, on peut prouver que toutes les structures de taille supérieure ou égale à $l_{max} = \lfloor h(s_0)/D_{c,TS} \rfloor$ auront forcément une heuristique moins bonne que celle de la plus petite structure consistante s_0 déjà trouvée, où $D_{c,TS}$ est le nombre de niveaux différents pour la compétence c présents dans TS et $\lfloor x \rfloor$ la partie entière par défaut de x.

Cette optimisation ne supprime pas le caractère exponentiel, mais permet de réduire fortement le nombre de structures à comparer. Des détails sur cet algorithme sont disponibles dans les articles [15] et [16].

Méthode de sélection

Les performances d'un algorithme d'apprentissage dépendent directement des informations présentes dans l'ensemble d'apprentissage, donc des exemples manuels qu'on lui fournit. Il est donc très important de disposer d'une bonne méthode pour sélectionner les copies les plus informatives possibles afin qu'elles soient ensuite évaluées manuellement par l'enseignant et minimiser le nombre d'exemples requis pour pouvoir prédire correctement les évaluations pour toutes les copies.

Nous avons choisi d'adopter une méthode de sélection par rejet. Elle fonctionne en deux temps : sélection des exemples manuels initiaux puis sélection d'une copie à évaluer manuellement à chaque itération du système.

Plus précisément, on va sélectionner initialement au hasard k copies qui formeront l'ensemble d'apprentissage initial TS une fois évaluées manuellement par l'enseignant pour la compétence c. A chaque itération, on va sélectionner aléatoirement une copie parmi toutes celles que l'algorithme n'a pas été capable

1. $\#Parties(QS) = 2^{\#QS}$

de classifier à l'itération précédente : une telle copie sera dite rejetée par l'algorithme. On va ensuite demander à l'enseignant d'évaluer manuellement cette copie pour la compétence c. Cette copie évaluée sera alors ajoutée à TS.

Cette méthode de sélection est volontairement très simpliste et pourra être améliorée dans l'avenir.

4 Expériences

Une fois implémenté en Java, nous avons voulu vérifier l'utilité et l'utilisabilité de notre système.

Nous avons mesuré l'utilité par le taux d'erreur de prédiction et le nombre de copies manuellement évaluées, et l'utilisabilité par la durée d'exécution par itération. Nous avons étudié ces critères dans plusieurs conditions expérimentales : fonction d'évaluation plus ou moins complexes, plus ou moins de copies sélectionnées initialement, plus ou moins de questions.

4.1 Expériences sur données simulées

Données simulées

Nous avons effectué nos expériences sur des données simulées. Pour créer ces données, nous avons généré $\#GS$ copies constituant les ensembles $PreTS$ et DS grâce à une distribution aléatoire uniforme sur 5 questions comportant chacune 5 réponses différentes possibles en s'assurant que ces deux ensembles soient disjoints. Nous avons également généré aléatoirement une fonction d'évaluation f_r dont on aura préalablement fixé la taille de la structure associée. Nous avons utilisé cette fonction afin de simuler l'évaluation manuelle de l'enseignant en évaluant 1 compétence à 4 niveaux possibles afin de former l'ensemble d'apprentissage TS, puis l'exemple supplémentaire à chaque itération. Nous l'avons également utilisé afin de vérifier l'exactitude des prédictions de l'algorithme.

Protocole de test

Nous avons testé 5 conditions expérimentales différentes : on a fait varier la taille $\#s_r$ de la structure de la fonction réelle, le nombre $\#TS_{init}$ d'exemples initialement fournis à l'algorithme d'apprentissage, le nombre total de copies $\#GS$ et le nombre de questions $\#QS$ comme décrit dans le tableau 2.

Tableau 2. Conditions expérimentales.

identifiant de l'expérience	$\#QS$	$\#s_r$	$\#TS_{init}$	$\#GS$
expérience A	5	1	10	100
expérience B	15	1	10	100
expérience C	5	3	10	100
expérience D	5	3	30	100
expérience E	5	1	10	89

A chaque itération du système nous avons noté le nombre de copies manuellement évaluées, le taux de performance de l'algorithme en comparant les prédictions de l'algorithme DS_p avec les évaluations réelles ainsi que la durée d'exécution de l'itération.

Par soucis de fiabilité des résultats nous avons répété chaque expérience 1000 fois et calculé la moyenne des résultats obtenus.

Résultats des expériences sur données simulées

Les diagrammes de la figure 1 présentent les performances moyennes de prédiction de notre système : taux moyen d'erreur, de rejet et de bonne prédiction en fonction du nombre de copies évaluées manuellement $\#TS_f$.

On a analysé ces résultats selon 4 axes : l'impact du nombre de questions $\#QS$, l'impact de la taille $\#s_r$ de la structure réelle, l'impact de la taille $\#TS_{init}$ de l'ensemble d'apprentissage à la première itération et l'impact du nombre total de copies $\#GS$ en comparant les expériences dont les conditions expérimentales ne varient que par un seul paramètre à la fois. On s'est plus particulièrement intéressé au taux final d'erreur de prédiction $\%err_f$, au nombre de copies utilisées comme exemples manuels à la fin du système $\#TS_f$.

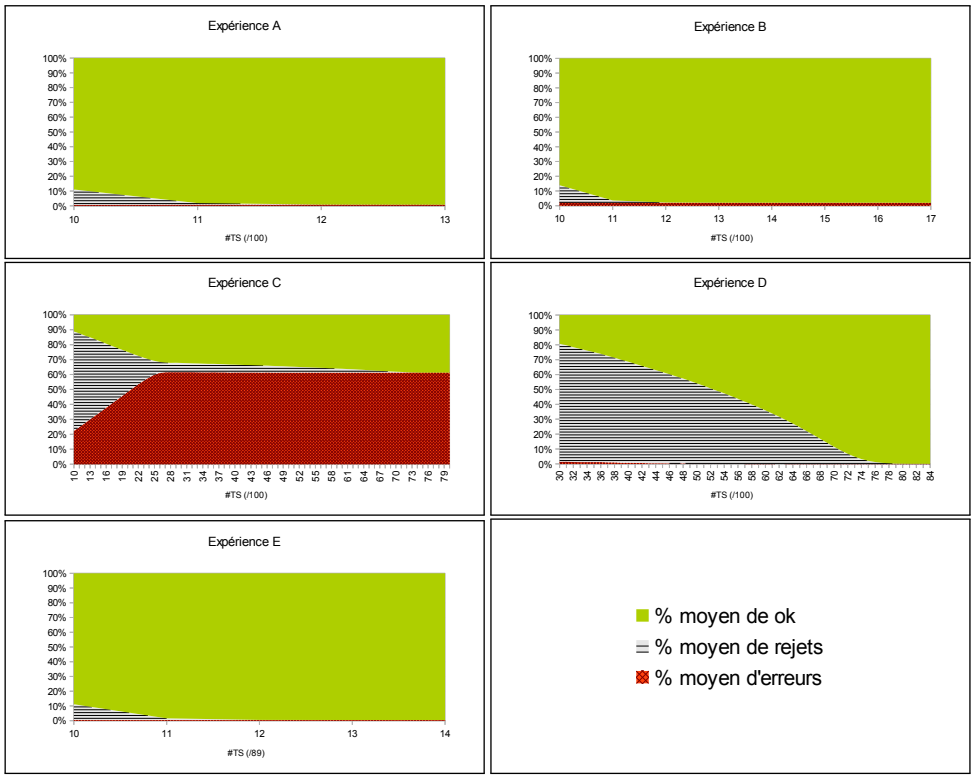

Fig. 1. Diagrammes pour les expériences sur données simulées

Pour analyser l'impact de la taille de la structure réelle $\#s_r$, on a comparé les résultats des expériences A et C. L'augmentation de la taille de $\#s_r$ semble s'accompagner d'une augmentation importante du taux final d'erreur et du nombre d'exemples manuels.

Pour analyser l'impact du nombre initial d'exemples manuels $\#TS_{init}$, on a comparé les résultats des expériences C et D. L'augmentation de $\#TS_{init}$ semble s'accompagner d'une diminution importante du taux d'erreur, jusqu'à un taux d'erreur nul pour les 1000 instances et une légère augmentation du nombre d'exemples manuels.

Pour analyser l'impact du nombre total de copies $\#GS$, on a comparé les résultats des expériences A et E. La diminution de $\#GS$ semble s'accompagner d'une légère diminution du taux d'erreur final, le nombre d'exemples manuels restant à peu près constant.

Pour analyser l'impact du nombre de questions $\#QS$, on a comparé les résultats des expériences A et B. L'augmentation du nombre de questions semble s'accompagner d'une légère augmentation du taux d'erreur final et du nombre d'exemples manuels utilisés.

Nous avons ensuite analysé les performances temporelles de notre système grâce aux données du tableau 3 qui présente la moyenne et l'intervalle de confiance de la durée d'exécution d'une itération du système.

Tableau 3. Durée moyenne d'exécution d'une itération du système avec intervalles de confiance (95%)

identifiant de l'expérience	durée d'exécution (en ms)
A	$0,407 \pm 0,038$
B	$0,678 \pm 0,050$
C	$0,572 \pm 0,025$
D	$2,008 \pm 0,018$
E	$0,430 \pm 0,041$

On remarque donc que :

- L'augmentation de la taille de la structure réelle semble s'accompagner d'une légère augmentation de la durée d'exécution par itération.
- Le nombre initial d'exemples manuels semble augmenter de manière importante la durée d'exécution par itération.
- La diminution du nombre total de copies a très peu d'impact sur la durée d'exécution par itération.
- L'augmentation du nombre de questions semble s'accompagner d'une augmentation significative de la durée d'exécution par itération.

Réflexion

Il semble donc possible de fournir initialement suffisamment d'exemples manuels au système pour garantir que le taux d'erreur final soit nul *i.e.* une fiabilité parfaite de prédiction, sous réserve que le nombre de questions et que la taille de la structure réelle ne soient pas trop élevés par rapport au nombre total de

copies. En contrepartie la durée d'exécution par itération semble augmenter de manière importante.

Nous avons donc réalisé une expérience sur des données réelles afin de tester cette hypothèse.

4.2 Expériences sur données réelles

Nous avons rassemblé nos données réelles en faisant passer à 89 apprenants de 1ère année de spécialité d'une école d'ingénieur un QCM sur Moodle portant sur les connaissances en programmation qu'ils ont acquis pendant les années précédentes. L'enseignant du cours a évalué *a posteriori* leurs réponses à l'aide d'une grille d'évaluation afin de leur attribuer un niveau pour chacune des 7 compétences différentes, et on s'est assuré qu'il n'y avait pas d'incohérence de notation entre les copies des apprenants. On présente ici les résultats pour la compétence la plus simple, « Identifier les instructions du schéma conditionnel »qui peut être associée à 3 niveaux : « Acquis »(73 apprenants), « En cours d'acquisition »(3 apprenants) et « Non acquis »(13 apprenants) et dont la structure ne dépend que de la réponse à 1 question sur les 29 que comportait le QCM.

Le protocole de test est le même que précédemment. Le diagramme 2 et le tableau 4 qui suivent présentent les résultats obtenus pour les expériences F :$\#TS_{init} = 10$ et G :$\#TS_{init} = 25$.

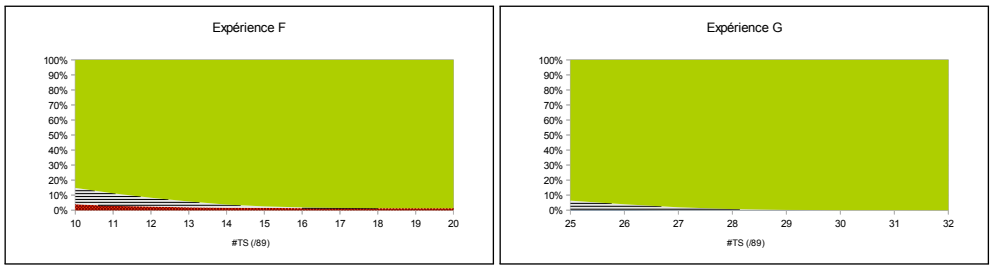

Fig. 2. Diagrammes pour les expériences sur données réelles

Tableau 4. Durée moyenne d'exécution d'une itération du système avec intervalles de confiance (95%)

identifiant de l'expérience	durée d'exécution (en ms)
expérience F	$1,335 \pm 0,037$
expérience G	$4,002 \pm 0,074$

Comme supposé, l'expérience G converge avec un taux d'erreur final nul, un nombre d'exemples manuels et une durée d'exécution par itération significativement plus grands que ceux de l'expérience G. Cependant les durées d'exécution par itération sont toutes inférieures à $5ms$. Sachant que les compétences peuvent être apprises séparément, le temps d'apprentissage est donc proportionnel au nombre de compétences. Cela s'avère donc déjà suffisamment court pour rester utilisable en pratique pour plus d'une vingtaine de compétence.

Il est à noter que ces résultats sont valables pour des compétences associées à des structures réelles de taille 1. Nous n'avons pas encore eu l'occasion de tester notre système avec des données réelles sur des compétences associées à des structures réelles de plus grande taille.

Nous sommes donc capables de garantir un taux d'erreur final nul avec suffisamment d'exemples initiaux dans ces conditions. La question se pose alors de comment calculer ce nombre en fonction des paramètres du problème : $\#QS$, $\#GS$ et $\#s_r$ ou à défaut, une borne maximale $s_{r,max}$.

5 Conclusion

En conclusion, nous avons décrit un système concret, utile et utilisable d'aide à l'évaluation simple et thématiquement indépendant. Nous avons analysé son utilité et son utilisabilité dans le cadre de différentes conditions expérimentales, ce qui nous a permis de comprendre que l'ajustement du nombre initial d'exemples manuels pouvait permettre de garantir un taux d'erreur nul sous réserve que la taille de la structure réelle et le nombre de questions ne soient pas trop élevés.

Il semble possible d'améliorer l'utilité de notre système en perfectionnant notre méthode de sélection afin de réduire le nombre initial d'exemples manuels, ce qui constitue un des problèmes du domaine de l'*Active Learning*. De plus d'après nos expériences il semble que cela pourrait également permettre de réduire la durée d'exécution de chaque itération de notre système. Il semble également pertinent d'estimer quantitativement jusqu'à quelle taille de structure réelle notre système d'aide peut répondre en temps raisonnable et ainsi rester utilisable ; autrement dit identifier les limites fonctionnelles de notre système.

Enfin, il apparaît possible de profiter du contexte des MOOC où les promotions sont bien plus larges, ce qui semble pouvoir naturellement limiter le taux d'erreur et amplifier le gain de temps potentiel. Dans ce cadre nous envisageons de porter notre système d'aide à l'évaluation en tant que plugin sur Moodle.

Références

1. Krathwohl, D. R. (2002). A revision of Bloom's taxonomy : An overview. Theory into practice, 41(4), 212-218.
2. Anderson, L. W., Krathwohl, D. R., Airasian, P. W., Cruikshank, K. A., Mayer, R. E., Pintrich, P. R., ... & Wittrock, M. C. (2001). A taxonomy for learning, teaching, and assessing : A revision of Bloom's taxonomy of educational objectives, abridged edition. White Plains, NY : Longman.
3. Rust, C., Price, M., & O'Donovan, B. (2003). Improving students' learning by developing their understanding of assessment criteria and processes. Assessment & Evaluation in Higher Education, 28(2), 147-164.
4. Conole, G., & Warburton, B. (2005). A review of computer-assisted assessment. ALT-J, 13(1), 17-31.

5. Demeuse, M. (2004). Introduction aux théories et aux méthodes de la mesure en sciences psychologiques et en sciences de l'éducation.
6. Beevers, C. E., & Paterson, J. S. (2003). Automatic assessment of problem-solving skills in Mathematics. Active Learning in Higher Education, 4(2), 127-144.
7. Christie, J. R. (1999, June). Automated essay marking-for both style and content. In Proceedings of the Third Annual Computer Assisted Assessment Conference, Loughborough University, Loughborough, UK.
8. POLANYI, M. (1998) The tacit dimension, in : L. PRUSAK (Ed.) Knowledge in Organization (Boston, MA, Butterworth Heineman).
9. Tsoukas, H. (1996). The firm as a distributed knowledge system : a constructionist approach. Strategic management journal, 17(WINTER), 11-25.
10. Leclercq, D. (1999). Édumétrie et docimologie. Liège : STE-ULG.
11. Butcher, P. G., & Jordan, S. E. (2010). A comparison of human and computer marking of short free-text student responses. Computers & Education, 55(2), 489-499.
12. Sangwin, C. J., & Grove, M. (2006, January). STACK : addressing the needs of the neglected learners. In Proceedings of the Web Advanced Learning Conference and Exhibition, WebALT (pp. 81-96).
13. Sangwin, C. J. (2007). STACK : making many fine judgements rapidly. In CAME 2007–the fifth CAME symposium (pp. 1-15).
14. Arneil, S., & Holmes, M. (1999). Juggling hot potatoes : decisions and compromises in creating authoring tools for the Web. ReCALL, 11(2), 12-19.
15. Follet, D., Delestre, N., Malandain, N., & Vercouter, L. (2013). Explicitation de connaissances tacites par évaluation assistée par ordinateur. Actes de la Journée EIAH&IA 2013.
16. Follet, D., Delestre, N., Malandain, N., & Vercouter, L. (2013, December). A three-step classification algorithm to assist criteria grid assessment. In NIPS 2013, Workshop Data Driven Education.