UNSL at eRisk 2019: a Unified Approach for Anorexia, Self-harm and Depression Detection in Social Media

Sergio G. Burdisso[1,2], Marcelo Errecalde[1], and Manuel Montes-y-Gómez[3]

[1] Universidad Nacional de San Luis (UNSL), Ejército de Los Andes 950, San Luis, San Lius, C.P. 5700, Argentina
{sburdisso, merreca}@unsl.edu.ar
[2] Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Argentina
[3] Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), Luis Enrique Erro No. 1, Sta. Ma. Tonantzintla, Puebla, C.P. 72840, Mexico
mmontesg@inaoep.mx

Abstract. In this paper we describe the participation of our research group at the CLEF eRisk 2019. The eRisk goal is the early detection of at-risk people by means of machine learning techniques based on language usage. This year eRisk edition was divided into three tasks, T1, T2, and T3. The first two were focused on early detection of anorexia and self-harm on Reddit users. T3 focused on measuring users' severity of depression. To carry out this task, models had to automatically fill the standard BDI depression questionnaire based on the evidence found in the user's history of postings. We used the same classifier, SS3, to carry out these three tasks with the same hyper-parameters configuration. SS3 is a recently introduced text classifier[1] that was created with the goal to deal with early risk detection scenarios in an integrated manner: it naturally supports incremental and early classification over text streams and additionally, it has the ability to visually explain its rationale. The final results for all these three tasks show that SS3 is a very robust and efficient classifier. SS3 was the fastest method and obtained the best $ERDE$ and overall best ranking-based measures in all the tasks. Additionally, it obtained the best $Precision$, $F1$ and $F1_{latency}$ for task T2. Finally, in task T3, it obtained the best AHR and ACR values, and the second-best ADODL and DCHR. This was quite remarkable taking into account that the same classifier was used here to fill users' BDI questionnaires, which is a task completely different from the other two "yes or no" tasks.

Keywords: SS3 · Early Risk Detection · Text Classification · Early Classification. · Text Streams Classification

1 Introduction

The detailed description of each task and the used performance measures, and full lists of results are given in [4]. Therefore, this paper will only focus on describing how we approached each task. All contributions we sent to the eRisk 2019 were implemented using a novel text classifier called SS3 which was recently introduced in [1]. SS3 was specially built to deal with early risk detection (ERD) tasks in an integrated manner since it naturally supports these 3 key aspects: (a) incremental training and classification; (b) early classification; and (c) having the ability to visually explain its rationale (i.e. provide the reasons for the classification). SS3 is a generalization of the classifier we used in the last year eRisk[3] for UNSLD and UNSLE runs. This year we decided not to use other models other than SS3 because of the change in the way data was released, i.e. a more realistic item-by-item release of data. The other models we used last year based on TVT[2] were no longer applicable since they were designed to work with chunks and not text streams. Additionally, this year we decided to put the robustness of SS3 into the test by using the same hyper-parameter configuration in all the 15 runs for the 3 tasks (T1, T2, and T3). Thus, the hyper-parameters values were set to $\lambda = \rho = 1$ and $\sigma = 0.455$, which were the same values used in [1], and for which SS3 has shown to be quite robust in terms of ERDE performance measure.

2 The SS3 Text Classifier

As it is described in more details in [1], SS3 first builds a dictionary of words for each category during the training phase, in which the frequency of each word is stored. Then, using those word frequencies, and during the classification stage, it calculates a value for each word using a function $gv(w, c)$ to value words in relation to categories. gv takes a word w and a category c and outputs a number in the interval [0,1] representing the degree of confidence with which w is believed to *exclusively* belong to c, for instance, suppose categories $C = \{food, music, health, sports\}$, we could have:

$gv(\text{`}sushi\text{'}, food) = 0.85$; $gv(\text{`}the\text{'}, food) = 0$;
$gv(\text{`}sushi\text{'}, music) = 0.09$; $gv(\text{`}the\text{'}, music) = 0$;
$gv(\text{`}sushi\text{'}, health) = 0.50$; $gv(\text{`}the\text{'}, health) = 0$;
$gv(\text{`}sushi\text{'}, sports) = 0.02$; $gv(\text{`}the\text{'}, sports) = 0$;

Additionally, a vectorial version of gv is defined as:

$$\overrightarrow{gv}(w) = (gv(w, c_0), gv(w, c_1), \ldots, gv(w, c_k))$$

where $c_i \in C$ (the set of all the categories). That is, $\overrightarrow{gv}$ is only applied to a word and it outputs a vector in which each component is the gv of that word for each category c_i. For instance, following the above example, we have:

$gv(\text{`}sushi\text{`}) = (0.85, 0.09, 0.5, 0.02)$; $gv(\text{`}the\text{`}) = (0, 0, 0, 0)$;

The vector $\overrightarrow{gv}(w)$ is called the "*confidence vector* of w". Note that each category c_i is assigned a fixed position in $\overrightarrow{gv}$. For instance, in the example above $(0.85, 0.09, 0.5, 0.02)$ is the *confidence vector* of the word "sushi" and the first position corresponds to $food$, the second to $music$, and so on.

It is worth mentioning that the computation of gv involves three functions, lv, sg and sn, as follows:

$$gv(w, c) = lv_\sigma(w, c) \cdot sg_\lambda(w, c) \cdot sn_\rho(w, c)$$

- $lv_\sigma(w, c)$ values a word based on the local frequency of w in c. As part of this process, the word distribution curve is smoothed by a factor controlled by the hyper-parameter σ.
- $sg_\lambda(w, c)$ captures the global significance of w in c, it decreases its value in relation to the lv value of w in the other categories; the hyper-parameter λ controls how far the local value must deviate from the median to be considered significant.
- $sn_\rho(w, c)$ sanctions lv in relation to how many other categories w is significant ($sg_\lambda(w, c) \approx 1$) to. That is, The more categories c_i whose $sg_\lambda(w, c_i)$ is high, the smaller the $sn_\rho(w, c)$ value. The ρ hyper-parameter controls how sensitive this sanction is.

For those readers interested in how these functions are actually computed, we highly recommend you to read the SS3 original paper[1], since the equations for lv, sg and sn are not given here to keep the present paper shorter and simpler. Note that using the gv function, it is quite straightforward for SS3 to visually justify its decisions if different blocks of the input are colored in relations to it, as can be seeing on an online demo available at http://tworld.io/ss3 in which users can try out SS3 for topic categorization. This is quite relevant when it comes to early detection tasks in which usually real people are involved, specialists should be able to manually analyze classified subjects and this type of visual tools could be really helpful to assist those specialists.

For all 3 tasks T1, T2 and T3 we carried out the classification of each user, incrementally as in [1]. That is, the used *summary operators* for all levels were the addition, .i.e $\oplus_j = addition$ for all j, which simplified the classification process to the summation of all words' $\overrightarrow{gv}$ vectors read so far, in symbols, for every subject s:

$$\overrightarrow{d_s} = \sum_{w \in WH_s} \overrightarrow{gv}(w) \tag{1}$$

where WH_s is the subject's writing history. Note that for all the tasks $\overrightarrow{d_s}$ was a vector with two components, one for the positive class and the other for the

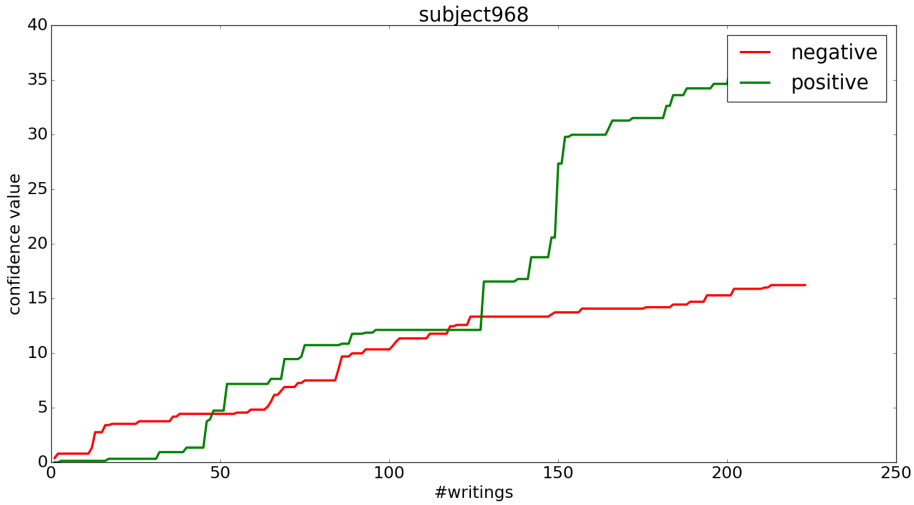


Fig. 1: Task 3's subject 968's positive and negative confidence value variation over time (writings).

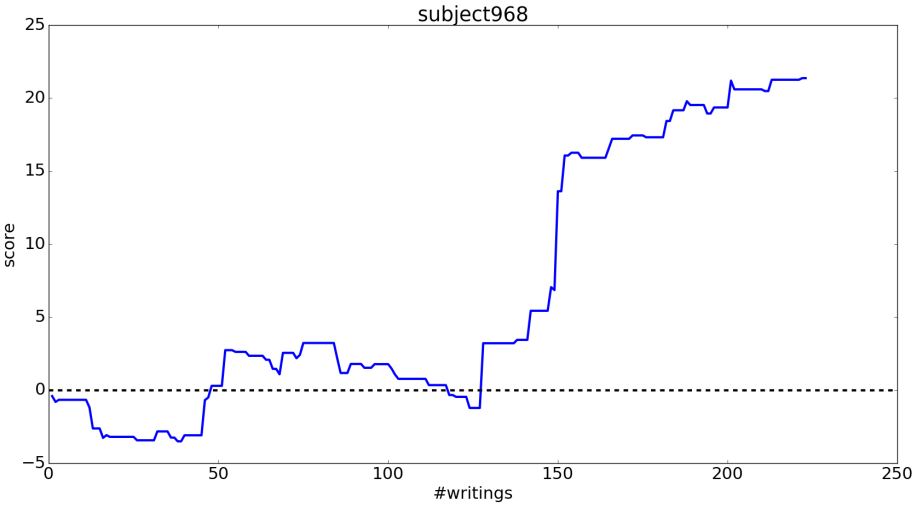


Fig. 2: Task 1's subject 968's estimated score of the level of anorexia ($d[positive] - d[negative]$) variation over time (writings).

negative one. The policy to classify a subject as positive was performed analyzing how $\overrightarrow{d_s}$ changed over time, as shown with an example in Figure 1. Subjects were

classified as positive when the positive value in $\overrightarrow{d_s}$ exceeded the negative one[4], for instance, the subject in the figure was classified as anorexic after reading the 42nd writing.

Finally, this year, models performance were also evaluated based on ranking-based measures for tasks T1 and T2. Thus, models were asked to provide an estimated score of the level of anorexia/self-harm along with the binary decision (0/1). To compute this score, SS3 performed the difference between the positive confidence value and the negative one (i.e. $d[positive] - d[negative]$) and returned it along with each decision. For example, in Figure 2 is shown how this score changed as more writings were read, for the same user shown previously in Figure 1.

3 Task 1: Early Detection of Signs of Anorexia

As mentioned earlier, we used the same classifier with the same hyper-parameters for these 5 runs, and instead, we mostly focused on changing aspects related to how we trained our models. For each one of the 5 runs, we performed the following:

- *UNSL#0*: we trained the model using only the data in the "train" folder, i.e. we trained the model using the same data as for the eRisk 2018.
- *UNSL#1*: the same as with the previous one but this time allowing SS3 to compute the *global value* not only for words but also for pair of words (bigrams), i.e. SS3 learned to compute $gv(w_0, c)$ as well as $gv(w_0 w_1, c)$ for each w_0, w_1 seen during training.
- *UNSL#2*: the same as run#0 but training with all available data for training this year, i.e. all the data in both "train" and "test" folders.
- *UNSL#3*: the same as in the previous run, but this time also taking into account bigrams of words (as in run#1).
- *UNSL#4*: the same as in run#2 but letting SS3 to take into account only words whose *global value* where greater than 0.3, i.e in Equation 1 SS3 assigned $gv(w, c_i) = 0$ to all w and c_i such that $gv(w, c_i) < 0.3$.

The main global results obtained in this task could be summarized as follows:

- As it is shown in Table 1, UNSL#0 obtained the best $ERDE_5$ and UNSL#4 the best $ERDE_{50}$. Note that most of the ERDE values are relatively close to each other, this is due to the way $ERDE$ was computed.[5] The larger and more unbalanced the dataset is, the asymptotically flatter and closer the

[4] Except for task T3 in which we did not perform an "early stop", and therefore every user was classified after processing the entire writing history.

[5] $cfp = \frac{\#positive\_users}{\#users}$ therefore each user, misclassified as false positive, added a value of $\frac{cfp}{\#users} = \frac{\#positive\_users}{\#users^2} = \frac{73}{815^2} = 0.0001$ to the final/reported $ERDE$ (and 0.001 in case of false negative or true positive after the o threshold).

Table 1: Results for Task 1, decision-based evaluation. Best results are also shown for comparison.

Team#run	P	R	$F1$	$ERDE_5$	$ERDE_{50}$	$F1_{latency}$	$F2_{latency}$
UDE#0	.51	.74	.61	8.48%	3.87%	.58	.64
lirmm#1	**.77**	.60	.68	9.09%	5.50%	.62	.57
lirmm#2	.66	.70	.68	9.24%	5.80%	.60	.60
Fazl#2	.09	**1**	.16	17.11%	11.22%	.14	.28
CLaC#0	.45	.74	.56	6.72%	3.93%	.54	.63
CLaC#1	.61	.82	.70	5.73%	3.12%	.69	**.75**
CLaC#2	.60	.81	.69	6.01%	3.13%	.68	.73
CLaC#3	.63	.81	.69	6.27%	3.54%	.68	.74
CLaC#4	.64	.79	**.71**	6.25%	3.42%	**.69**	.73
LTL-INAOE#1	.47	.75	.58	7.74%	4.19%	.55	.64
INAOE-CIMAT#0	.56	.78	.66	9.29%	3.98%	.62	.68
INAOE-CIMAT#3	.67	.68	.68	9.17%	4.75%	.63	.69
INAOE-CIMAT#4	.69	.63	.66	9.12%	5.07%	.61	.59
UNSL#0	.42	.78	.55	**5.53%**	3.92%	.55	.66
UNSL#1	.43	.75	.55	5.68%	4.10%	.55	.65
UNSL#2	.36	.86	.51	5.56%	3.34%	.50	.67
UNSL#3	.35	.85	.50	5.59%	3.48%	.49	.66
UNSL#4	.31	.92	.47	6.14%	**2.96%**	.46	.65

Table 2: Participating teams for Task 1: number of runs, number of user writings processed by the team, and the time taken for the whole process and for processing each writing (i.e $\frac{TotalTime}{\#writings \times \#runs}$). (*) This value was originally 5, but we replaced it by the actual number of models used.

			Time	
Team	#runs	#writings	Total	Per writing
UppsalaNLP	5	2000	2d 7h	20s
BioInfo@UAVR	1	2000	14h	25s
BiTeM	4*	11	4h	5m 30s
lirmm	5	2024	8d 15h	1m 12s
CLaC	5	109	11d 16h	31m
SINAI	3	317	10d 7h	15m 36s
HULAT	5	83	18h	2m 36s
UDE	3*	2000	5d 3h	1m 12s
SSN-NLP	5	9	6d 22h	3h 42m
Fazl	3	2001	21d 15h	5m 12s
UNSL	5	2000	23h	**8s**
LTL-INAOE	2	2001	17d 23h	6m 30s
INAOE-CIMAT	4*	2000	8d 2h	1m 30s

$ERDE$ values are (as it was with this task). Thus, decimals do matter a lot when it comes to $ERDE$ measure. For instance, in this task, just a small difference of 0.009 (0.9%) in $ERDE$ actually means that 12% of either all

Table 3: Ranking-based evaluation results for Task 1. Here it is shown the best results for SS3, UNSL #2 and #4, along with the other 3 best ones, LTL-INAOE and UDE #0 and #1.

	1 writing			100 writing		
Team#run	P@10	NDCG@10	NDCG@100	P@10	NDCG@10	NDCG@100
UNSL#2	**.8**	***.82***	***.55***	**1**	**1**	.83
UNSL#4	**.8**	***.82***	.52	.9	.94	***.85***
LTL-INAOE#0	**.8**	.75	.34	**1**	**1**	.76
UDE#0	.2	.12	.11	.9	.92	.81
UDE#1	.6	.75	.54	.9	.94	.81
	500 writing			1000 writing		
Team#run	P@10	NDCG@10	NDCG@100	P@10	NDCG@10	NDCG@100
UNSL#2	**1**	**1**	.83	**1**	**1**	*.84*
UNSL#4	**1**	**1**	*.85*	.9	.94	*.84*
LTL-INAOE#0	.9	.92	.73	.7	.78	.65
UDE#0	.9	.93	*.85*	.9	.94	.86
UDE#1	**1**	**1**	**.87**	**1**	**1**	**.88**

anorexic users or non-anorexic ones were not properly classified, which is a significant difference.

- Regarding the new $F1_{latency}$, we did not obtain remarkable results, being 0.13 points below the best one. This is mainly due to this new measure being introduced this year and only after the tasks ended. Therefore, SS3 could not be optimized to obtain a better $F1_{latency}$ value. As said before, we used the same hyper-parameters for the 15 runs of the 3 tasks, these hyper-parameters were selected to optimize ERDE measures, which in turns produced an SS3 model that prioritizes the recall[6] and speed[7] above the precision, which is not bad taking into account that we are dealing with early risk detection tasks (every positive subject not detected is a life at risk!). Despite this, our best $F1_{latency}$ value (.55) was quite above the average (0.38) and was positioned 11th out of the 50 contributions and 5th out of the 13 research groups. Additionally, we also decided to compute the $F2_{latency}$ which gives a little more of importance to recall than to precision. This improved our results, making our best $F2_{latency}$ value (.67) to be positioned 7th out of the 50 contributions and 3rd out of the 13 research groups, and only 0.07 points below the best one.
- As it is shown in Table 2, SS3 was the fastest method to process the writings from each server response, processing each users' writing in about 8s[8], this

[6] This is due to the way ERDE measure is computed, the false positive cost (cfp) is really small compared to the false negative cost (cfn).

[7] On average, SS3 classified users after reading the 2nd or 3rd post.

[8] Note that much of this 8s were wasted waiting for network communication, since this number includes the latency of receiving and sending the response from and to the API RESTful server.

contrast with other methods that obtained a better $F1_{latency}$ but required more time, such as CLaC, INAOE-CIMAT or lirmm. For instance, CLaC was 232 times slower than SS3 and took 11 days and 16h to process only 109 of the 2000 writings, whereas SS3 processed all the 2000 writings for the 5 runs in only 23h. This could suggest that some research groups possible incorporated some sort of offline (or manual) processing into their models. It is worth mentioning that the fact that SS3 was the fastest model was not due to the type of machine we used[9] but rather due to SS3 naturally supporting *incremental classification*. To put this point in context, it is important to note that ERD is essentially a problem of *analysis of sequential data*. That is, unlike traditional supervised learning problems where learning and classification are done on "complete" objects, here classification must be done on "partial" objects which correspond to all the data sequentially read up to the present, from a (virtually infinite) data stream. Algorithms capable of naturally dealing with this scenario are said to support *incremental classification*. As it is described in more details in [1], unlike most state-of-the-art classifiers, SS3 supports *incremental classification* since it does not necessarily "see" the input stream as an atomic n-dimensional vector (i.e. a document vector) that must be computed entirely before making a prediction. In consequence, when working with a sequence of documents, common classifiers must re-compute the input vector each time new content is added to the sequence[10]. Formally, if n is the length of the stream/sequence of items, when working with SS3, the cost of the early classification algorithm *for every subject*, according to the number of processed items, is equal to n (since each item needs to be processed only once). On the other hand, for classifiers not supporting incremental classification (such as SVM, LOGREG, KNN or any type of non-recurrent Neural Networks), this cost is equal to $n \times (n+1)/2 = 1 + 2 + ... + n$ (since the first item needs to be processed n times, the second $n-1$, the third $n-2$, and so on). Thus, we have classifiers supporting stream classifications, such as SS3, belonging to $O(n)$ whereas the others to $O(n^2)$.

– SS3 was the method that obtained the best overall performance in ranking-based evaluation since, as shown in Table 3: it obtained the best ranking performance P@10 and NDCG@10 for all the 4 rankings; the best NDCG@100 for rankings made after processing 1 and 100 writings (.55 and .85 respectively); and additionally, for the ranking made after processing 500 obtained the second-best NDCG@100 (.85, first was .87) and the third-best NDCG@100 (.84, first was .88) for the ranking made after processing 1000

[9] We coded our script in plain python 2.7 and only using built-in functions and data structures, no external library was used (such as *numpy*). Additionally, to run our script we used one of the author's personal laptop which had standard technical specifications (Intel Core i5, 8GB of DDR4 RAM, etc.).

[10] Since the "input document" is a stream, the input is a "document" that grows over time!

Fig. 3: Top-100 words selected by *global value (gv)* from the model trained for the task T1. Words are sized by *gv*.

writings.[11] Note that these results are not a minor aspect, since they are implying that both: (a) the score (*confidence value*) given by SS3 correctly values/ranks positive subjects, that is, the *global value*, *gv*, is correctly capturing the degree of importance of each word for the positive class[12] (see Figure 3 for a top-100 word cloud selected by *gv*); and (b) since SS3 is valuing/ranking users correctly, it means there is much room for improving the classification performance by choosing a better policy to actually classify them —perhaps using global information across different users could lead us to better classification performance, instead of classifying them locally, simply and prematurely just when the positive value exceeds the negative one.

[11] Those first NDCG@100 values, .87 and .88, were obtained by UDE#1. It is worth mentioning that it took UDE 1 day and 6h to process those 500 writings (or 2 days and 12h for those 1000 writings) whereas it took SS3 only 5h to process them (9 times faster).

[12] which, as we will see later, this is also reflected by the promising results obtained in task T3.

Table 4: Results for Task 2, decision-based evaluation. Best results are also shown for comparison.

Team#run	P	R	$F1$	$ERDE_5$	$ERDE_{50}$	$F1_{latency}$	$F2_{latency}$
BiTeM#0	.52	.41	.46	9.73%	7.62%	.46	.43
Fazl#2	.12	**1**	.22	22.66%	13.23%	.19	.35
UNSL#0	**.71**	.41	**.52**	9.00%	7.30%	**.52**	.45
UNSL#1	.70	.39	.50	9.02%	7.60%	.50	.43
UNSL#2	.20	.90	.32	9.19%	6.86%	.32	.53
UNSL#3	.31	.85	.45	8.78%	5.44%	.45	.63
UNSL#4	.31	.88	.46	**8.20%**	**4.93%**	.45	**.64**

4 Task 2: Early Detection of Signs of Self-harm

For this task, unlike T1, the training set was not provided, and therefore we had to build our own dataset to train SS3. To achieve this, we tried out creating different datasets, for instance, collecting tweets and Reddit posts related to self-harm, or using the datasets already available for anorexia and depression, as it is described in more details below:

- *UNSL#0*: we collected Reddit posts related to self-harm and stored them in a single txt file to represent the positive class. For the negative class, we used the negative documents in the "train" folder for task T1 (anorexia). This run obtained the best precision (.71) but, among the other 4 runs, the lowest recall (.41) along with UNSL#1 (.39), both using the same dataset.
- *UNSL#1*: this run used the same dataset as the previous one, but this time SS3 took into account also bigrams of words (as in UNSL#1 for T1).
- *UNSL#2*: for this run, we trained SS3 using a dataset built using the Reddit posts (the same used in the runs above) and tweets related to self-harm. We created a single file with all these tweets and posts related to self-harm (about 40MB in size) and used it to learn the positive class. For the negative class, we used the negative training documents for the eRisk 2018 depression task. Additionally, as in run#4 of T1, SS3 was configured to ignore words whose *global value* was less than 0.3. Among the other 4 runs, this one had the best Recall (.9) but the worst values for precision (.2) and F1 (.32).
- *UNSL#3*: here we trained SS3 using the training documents for T1 (anorexia) and also using the training documents for the eRisk 2018 depression task. This run had a similar performance to run#4, although its recall was a little bit worse.
- *UNSL#4*: SS3 was trained using the same documents as in the previous run (i.e. anorexia + depression 2018) but this time adding those of run#0. This run had the best $F2_{latency}$ (.64) and $ERDE$ values, 8.20% and 4.93% for $ERDE_5$ and $ERDE_{50}$ respectively.

Since no training data was released, we did not have any validation set to check if our models were learning properly, i.e. we did not know on which data

(a) Words (b) Word bigrams

Fig. 4: Top-100 words and word bigrams selected by *global value (gv)* from the model trained for the task T2. Both are sized by *gv*.

Table 5: Participating teams for Task 2: number of runs, number of user writings processed by the team, and the time taken for the whole process and for processing each writing (i.e $\frac{TotalTime}{\#writings \times \#runs}$). (*) This value was originally 5, but we replaced it by the actual number of models used.

			Time	
Team	#runs	#writings	Total	Per writing
BiTeM	2*	8	3m	11.2s
BioInfo@UAVR	1	1992	4h	7.2s
Fazl	3	1993	18d 21h	272s
UNSL	5	1992	13h	**4.6s**
UDE	4*	1992	1d 2h	11.7s
LTL-INAOE	4	1993	17h	7.6s
lirmm	5	2004	2d 22h	25.1s
CAMH	5	1992	1d 19h	15.5s

our models were going to be evaluated. In order to know whether the learned model made sense or not, after training, we asked SS3 to give us a list of words ordered by *global value* for the positive class, and checked if the list made sense to us. Fortunately, as shown in Figure 4, the generated list of words matched what we expected.

The global results obtained in this task could be summarized as follows:

Table 6: Ranking-based evaluation results for Task 2. Here it is shown the best results for SS3 along with the other best one, Fazl#1.

	1 writing			100 writing		
Team#run	P@10	NDCG@10	NDCG@100	P@10	NDCG@10	NDCG@100
UNSL#0	.7	.79	.48	**.9**	**.94**	.61
UNSL#1	.6	.74	.48	**.9**	**.94**	.60
UNSL#3	**1**	**1**	**.67**	**.9**	**.94**	.84
UNSL#4	**1**	**1**	.64	**.9**	.93	**.86**
Fazl#1	.2	.27	.36	**.9**	**.94**	.83
	500 writing			1000 writing		
Team#run	P@10	NDCG@10	NDCG@100	P@10	NDCG@10	NDCG@100
UNSL#0	**.9**	**.94**	.66	**.9**	**.94**	.66
UNSL#1	**.9**	**.94**	.65	**.9**	**.94**	.65
UNSL#3	.7	.63	.75	.7	.63	.75
UNSL#4	.7	.67	*.79*	.8	.74	*.78*
Fazl#1	**.9**	**.94**	**.84**	**.9**	**.94**	**.84**

- As it is shown in Table 4, UNSL#0 obtained the best precision (.71), $F1$ (.52) and $F1_{latency}$ (.52), and UNSL#4 the best $ERDE_5$ (8.20%) and $ERDE_{50}$ (4.93%).
- Once again SS3 was the fastest method, processing all the writing of each response in about 5s (as shown in Table 5).
- Again, SS3 was the method that obtained the best overall performance in ranking-based evaluation since, as shown in Table 6: it obtained the best ranking performance P@10 and NDCG@10 for all the 4 rankings; the best NDCG@100 for rankings made after processing 1 and 100 writings (.67 and .86 respectively). Additionally, for the ranking made after processing 500 and 1000 writings, SS3 obtained the second-best NDCG@100 (.79 and .78 respectively), the first ones were obtained by Fazl#1 (.84). It is worth mentioning it took Fazl 4 days and 17h to process those 500 writings (or 9 days and 10h for the 1000 writings), whereas it took SS3 only 3h (almost 60 times faster!).

It is worth mentioning that, unlike the other runs, UNSL#3 was trained only using data from anorexia and depression and yet it obtained good results. In fact, if we had sent only this run, among all participants, SS3 would have still obtained the best ERDE values (8.78% and 5.44%), the best $F2_{latency}$ (.63) and the second best $F1_{latency}$ (.45, first would have been .46). This, added to the results obtained by the other 4 runs, shows us that SS3 is a classifier quite robust to deal with cross-domain scenarios.

5 Task 3: Measuring the severity of the signs of depression

This task was really difficult since it was not a single "yes or no" problem but a problem involving multiple decisions, one for each one of the 21 questions. To make things even harder, as with Task 2, no training data was released either. Fortunately, early depression detection is a task we had some previous experience working with since we had participated in the two previous eRisk labs (2017[1] and 2018[3]). Therefore, we decided to train SS3 using the dataset for the eRisk 2018 depression detection task. However, the main problem was deciding how to turn this "yes or no" classifier into a classifier capable of filling BDI questionnaires. We came up with the idea of using the *confidence vector*, $\overrightarrow{d}$ in Equation 1, to somehow infer a BDI depression level between 0 and 63. To achieve this, first, we converted the *confidence vector* into a single *confidence value* normalized between 0 and 1, by applying the following equation:

$$confidence\_value = \frac{d[positive] - d[negative]}{d[positive]} \tag{2}$$

Then, after SS3 classified a subject, the obtained $confidence\_value$ was divided into 4 regions, one for each BDI depression category. This was carried out by the following equation:

$$c = \lfloor confidence\_value \times 4 \rfloor \tag{3}$$

And finally, the subject depression level was predicted by mapping the percentage of $confidence\_value$ left inside the predicted c region to its corresponding BDI depression level range (e.g. $(0.5, 0.75] \longrightarrow [19, 29]$ for $c = 2 =$ "moderate depression") by computing the following:

$$depression\_level = min_c + \lfloor (max_c - min_c + 1) \times (confidence\_value \times 4 - c) \rfloor \tag{4}$$

Where min_c and max_c are the lower and upper bound for category c, respectively (e.g. 19 and 29 for "moderate depression" category).

In order to clarify the above process, we will illustrate it with the example shown in Figure 5. First, SS3 processed the entire writing history computing the $confidence\_value$ (given by Equation 2) and then, the final $confidence\_value$ (0.941) was used to predict the depression category, "severe depression" ($c = 3$), by using the Equation 3. Finally, the depression level was computed by the mapping given by Equation 4, as follows:

$$\begin{aligned} depression\_level &= 30 + \lfloor (63 - 30 + 1) \times (0.941 \times 4 - 3) \rfloor \\ &= 30 + \lfloor 34 \times (3.764 - 3) \rfloor = 30 + \lfloor 34 \times 0.764 \rfloor \\ &= 30 + 25 = \mathbf{55} \end{aligned} \tag{5}$$

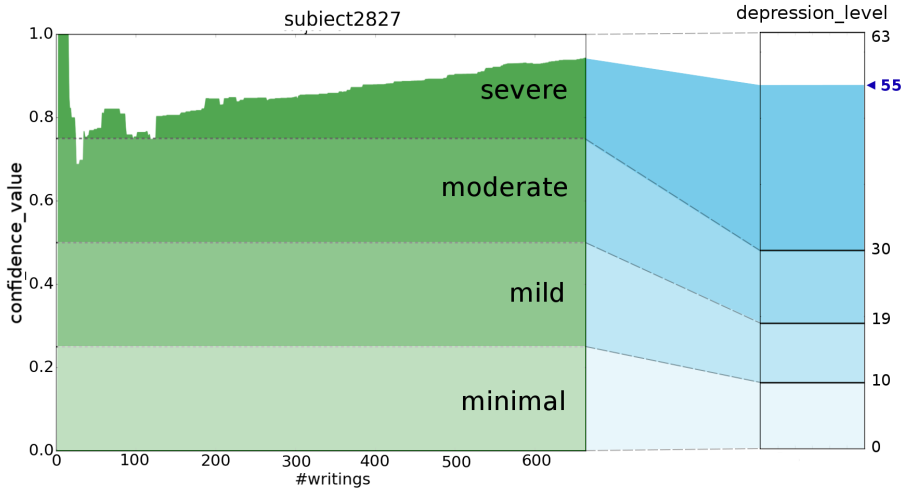
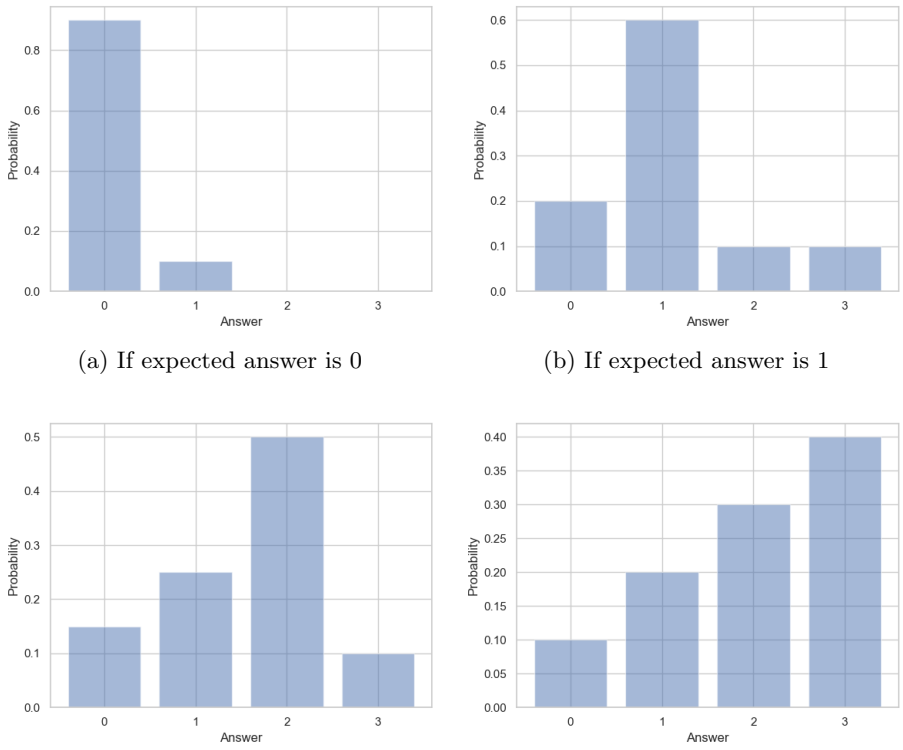


Fig. 5: Diagram of the *depression_level* computation process for subject 2827. As reader can notice, after processing all the subject's writings, the final *confidence_value* (0.941) was mapped into its corresponding *depression_level* (55).

At this point, we have transformed the output of SS3 from a 2-dimensional vector, d, into a BDI depression level (a value between 0 and 63). However, we have not covered yet how to actually answer the 21 questions in the BDI questionnaire using this *depression level*. Regardless of the method, we decided that for all those users whose *depression_level* was less or equal to 0, all the BDI questions were answered with 0. For the other users we applied different methods, depending on the run, as described below:

- *UNSLA*: using the predicted *depression_level* our model filled the questionnaires answering the expected number ($\lfloor \frac{depression\_level}{21} \rfloor$) on each question. If this division had a remainder, the remainder points were randomly scattered so that the sum of all the answers always matched the predicted depression level given by SS3.
- *UNSLB*: this time, only the predicted category, c, was used. Our model filled the questionnaire randomly in such a way that the final depression level always matched the predicted category. Compared to the following three ones, these two models were the ones with the worst performance.
- *UNSLC*: this model and the followings were more question-centered. Once again, as in UNSLA, our model filled the questionnaires answering the expected number derived from the predicted depression level ($\lfloor \frac{depression\_level}{21} \rfloor$). But this time, answering this number only on questions for which a "textual hint" for a possible answer was found in the user's writings, and randomly and uniformly answered between 0 and $\lceil \frac{depression\_level}{21} \rceil$ otherwise. To find this "textual hint", our model split the user's writings into sentences and searched for the co-occurrence of the word "I" or "my" with at least one



(a) If expected answer is 0

(b) If expected answer is 1

(c) If expected answer is 2

(d) If expected answer is 3

Fig. 6: Discrete probability distribution for each possible expected answer.

word matching a regular expression specially crafted for each question.[13] This method obtained the best AHR (41.43%) and the second-best DCHR (40%).

– *UNSLD*: the same as the previous one, but not using the "textual hints", i.e. always answering every question randomly and uniformly between 0 and $\lceil \frac{depression\_level}{21} \rceil$. This model was mainly used only with the goal of measuring the actual impact of using these "textual hints" to decide which questions should be answered with the expected answer ($\lfloor \frac{depression\_level}{21} \rfloor$).
– *UNSLE*: the same as UNSLD, but this time not using a uniform distribution. More precisely, from the overall depression level predicted by SS3, once again the expected answer was computed ($\lfloor \frac{depression\_level}{21} \rfloor$) and, depending on the value of the expected answer, actual answers were given following the probability distributions shown in Figure 6. Note that, unlike uniform

[13] e.g. "(sad)|(unhappy)" for question 1, "(future)|(work out)" for question 2, "fail\w*" for question 3, "(pleasure)|(enjoy)" for question 4, etc.

Table 7: Results for Task 3. Best results are also shown for comparison.

run	AHR	ACR	ADODL	DCHR
CAMH_GPT_nearest_unsupervised	23.81%	57.06%	**81.03%**	**45%**
UNSLA	37.38%	67.94%	72.86%	30%
UNSLB	36.93%	70.16%	76.83%	30%
UNSLC	**41.43%**	69.13%	78.02%	*40%*
UNSLD	38.10%	67.22%	78.02%	30%
UNSLE	40.71%	**71.27%**	*80.48%*	35%

Table 8: Results for Task 3. Now our runs results are shown using a 95% confidence interval.

run	AHR	ACR	ADODL	DCHR
UNSLA	38.13% ± 2.29%	68.30% ± 0.84%	72.62%	30%
UNSLB	38.97% ± 2.65%	**69.77% ± 1.98%**	74.72% ± 2.82%	30%
UNSLC	**40.19% ± 3.14%**	69.26% ± 1.75%	77.53% ± 1.92%	29.91% ± 8.70%
UNSLD	39.26% ± 3.21%	68.72% ± 1.81%	77.56% ± 1.86%	30.86% ± 11.14%
UNSLE	38.18% ± 3.41%	69.61% ± 1.95%	**82.94% ± 2.17%**	**38.42% ± 10.14%**

distribution (used in UNSLD), when using these probability distributions the expected answer is more likely to be selected over the other ones. This model obtained the best ACR (71.27%) and the second-best AHR (40.71%) and ADODL (80.48%, best was only 0.54% above).

The obtained results are shown in Table 7. As mentioned above, we obtained the best AHR (41.43%) and ACR (71.27%), and the second-best ADODL (80.48%) and DCHR (40%). However, since most of our models' answers are randomly generated, it implies that all of these measures are also stochastically generated.[14] The natural question in cases like this is "How do we know these results properly represent our models' performance and we did not obtain them just by pure chance?". In order to clarify this, we run each model 1000 times and calculated the values for AHR, ACR, ADODL and DCHR each time[15]. After this process finished, we ended up with a sample of 1000 values for each measure and model, which we then used to produce the results shown in Table 8. Results have been replaced by intervals with 95% of confidence, which better represent our performance. One can notice that, in fact, when we participated we had a little bit of bad luck, especially for UNSLE's ADODL, because the actual value we obtained (80.84%) is almost a lower bound outlier. Another thing that we can notice, comparing UNSLC and UNSLD, is that the use of "textual hints" slightly improves the Average Hit Rate (AHR) but does not impact on the other measures. UNSLE is considerably the best method to estimate the overall depression level since it takes values within a range that is quite above the

[14] Only ADODL and DCHR for UNSLA and DHR for UNSLB are deterministically determined by $depression_level$ and c.

[15] Just as if we had participated 1000 times in this task.

others. Additionally, another important point is that taking into account these 95% confidence intervals, the obtained values would be among the best ones even in the worst cases. Finally, since all the methods we used are based on the depression level predicted by SS3, this shows us that SS3 is correctly inferring the depression level from the textual evidence accumulated while processing the user's writings, i.e. SS3 is correctly valuing words in relation to each category (depressed and non-depressed) which is consistent with the results obtained for the ranking-based measures for task T1 and T2. Additionally, this could also imply that could really be a relationship between how subjects write (what words they use) and the actual depression level they have.

6 Conclusions and Future Work

In this article, we described the participation of our research group[16] at the CLEF eRisk 2019[4]. We described how we approached each one of the three tasks using the same SS3 classifier with the same hyper-parameter configuration. We showed how we mostly focused on aspects related to how we trained this classifier to create the different runs. For example, in task T2 we described for every run what data we used to train our model with. For this task, we also highlighted the cross-domain robustness that SS3 showed by the final results, in particular for UNSL#3 that obtained quite good performance despite being trained with data from anorexia and depression. For task 3, we described how we converted SS3 into a model capable of predicting a BDI depression level (from 0 to 63) which was later used to fill the questionnaires using different methods. The final results for all these three tasks showed that SS3 is a very robust and efficient classifier. SS3 was the fastest method and obtained the best $ERDE$ and the overall best ranking-based measures in all the tasks. Additionally, it obtained the best $Precision$, $F1$ and $F1_{latency}$ for task T2. In task T3, it obtained the best AHR and ACR values, and the second-best ADODL and DCHR. The results obtained for this task, along with those based on ranking measures, showed us strong evidence that SS3 properly values words in relation to how relevant they are to each category and therefore, the final *confidence value* properly values the text created by the subjects. Finally, overall results showed us that SS3 is a robust method since it obtained a remarkable overall performance in the three tasks despite using the same hyper-parameter configuration. For future work, we plan to mainly focus on three aspects. Given the interesting nature and implications of results in task T3, we will analyze in more details the obtained results, including a more qualitative analysis in which individual subjects could be analyzed. Additionally, we will explore different variations to improve the predicted depression level. Regarding task T1, we will explore different hyper-parameter values to improve the performance in terms of the new $F1_{latency}$ measure. Finally, based on the good results obtained for ranking-based measures, we plan to design better early classification policies in the future. Current policy tends to be "too hasty" so, we hope that delaying the decision until

[16] From the Universidad Nacional de San Luis (UNSL), San Luis, Argentina.

there is "enough confidence" to correctly classify subjects along with the use of global information across all the subjects could help to improve classification performance.

References

1. Burdisso, S.G., Errecalde, M., y Gómez, M.M.: A text classification framework for simple and effective early depression detection over social media streams. Expert Systems with Applications **133**, 182 – 197 (2019). https://doi.org/10.1016/j.eswa.2019.05.023, http://www.sciencedirect.com/science/article/pii/S0957417419303525
2. Errecalde, M.L., Villegas, M.P., Funez, D.G., Ucelay, M.J.G., Cagnina, L.C.: Temporal variation of terms as concept space for early risk prediction. In: CLEF (Working Notes) (2017)
3. Funez, D.G., Ucelay, M.J.G., Villegas, M.P., Burdisso, S.G., Cagnina, L.C., Montes-y Gómez, M., Errecalde, M.L.: Unsls participation at erisk 2018 lab
4. Losada, D.E., Crestani, F., Parapar, J.: Overview of eRisk 2019: Early Risk Prediction on the Internet. In: Experimental IR Meets Multilinguality, Multimodality, and Interaction. 10th International Conference of the CLEF Association, CLEF 2019. Springer International Publishing, Lugano, Switzerland (2019)