Detecting Packed Executable Files

Ivan Čeh
Sebastijan Horvat

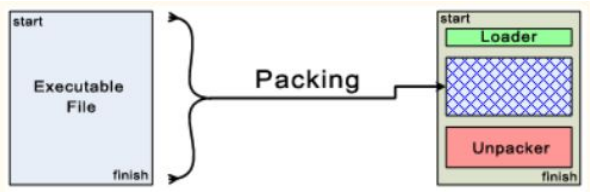
Sažetak

U ovom radu izradili smo klasifikator za odvajanje pakiranih od nepakiranih datoteka koji bi mogao biti korišten za ubrzavanje prepoznavanja malicioznih datoteka. Tvrtke *ReversingLabs* u sklopu natjecanja Mozgalo 2018. dala je bazu datoteka koje su klasificirane kao pakirane ili nepakirane. Značajke smo vadili iz pripadnog *TitaniumCore* izvještaja za pojedinu datoteku, koje nam je *ReversigLabs* također dao u sklopu natjecanja. Isprobali smo nekoliko modela strojnog učenja, a najboljima su se pokazale slučajne šume. Njima smo na testnom skupu, koji je sadržavao i do sad neviđene tipove pakiranih datoteka, postigli točnost od 97.98%.

Uvod

Pakiranje je metoda izmjene izvršivih datoteka bez mijenjanja njihove izvorne funkcionalnosti, ali na način da se datoteka zaštiti od reverznog inženjeringa, da se smanji veličina originalne izvršive daoteke, ili da se prikrije maliciozni izvršivi kod. Alate koji proizvode pakirane datoteke nazivamo pakerima. Za danu datoteku paker mijenja njen sadržaj, te dodaje instrukcije koje će taj sadržaj prilikom izvršavanja obnoviti. To se može vidjeti na slici 1.1. Dakle, pakirana datoteka sadrži enkriptiranu verziju originalne datoteke i postupak dekripcije, a izvršavanjem takve datoteke dekriptira se i pokreće izvršavanje originalne datoteke.

Mijenjanjem sadržaja datoteke, posebno se mijenja i potpis kod malicioznih datoteka, što antivirusnim alatima onemogućava njihovo prepoznavanje uobičajenim načinom preko prepoznavanja potpisa. Prema literaturi [1], 2006. godine preko 92% svih malicioznih datoteka bilo je pakirano. Pakiranje je osobito popularno među širiteljima malicioznih programa jer je dovoljno istu datoteku svaki puta pakirati novim pakerom da bi se izbjeglo otkrivanje od strane anitivirusnog programa. Česta je pojava i višestuko pakiranih datoteka (dakle pakiranje već pakiranih datoteka).



Slika 1.1. Prikaz procesa pakiranja

Univerzalno rješenje za otkrivanje malicioznih datoteka koje su pakirane dobiva se primjenom univerzalnih otpakirača. Univerzalni otpakirači mogu otkriti i dešifrirati kod pakiranih izvršivih datoteka. Jednom kada su dešifrirane, maliciozne datoteke se mogu otkriti tradicionalnim antivirusnim softverima kroz prepoznavanje potpisa. No takvi otpakirači su izuzetno računski skupi (zbog velike složenosti). Za veće kolekcije datoteka, otkrivanje malicioznih datoteka može trajati satima ili čak danima. Stoga su potrebne tehnike prepoznavanja obrazaca

za brzo otkrivanje pakiranih datoteka, tako da, jednom kada efektivno razdvojimo pakirane od nepakiranih, univerzalni otpakirač možemo pokrenuti na onima koje su označene kao pakirane.

Glavni cilj ovog projekta je razviti klasifikator koji će uspješno raspoznavati pakirane od nepakiranih datoteka. Time bi se relativno brzo dobila podjela na pakirane i nepakirane datoteke, čime bi se smanjio broj datoteka na kojima se pokreće univerzalni otpakirač. Datoteke koje se proglašene nepakiranima izravno bi se proslijeđivale antivirusnim alatima koji bi primijenili standardno prepoznavanje preko otkrivanja potpisa.

Prikupljanje podataka

U sklopu natjecanja Mozgalo 2018. tvrtka *ReversingLabs* dala je ukupno 43784 izvršivih datoteka. Za svaku datoteku dana je i vrijednost binarne ciljne varijable. Pritom je vrijednost ciljne varijable 1 ukoliko se radi o pakiranoj datoteci, a vrijednost je 0 ukoliko se radi o nepakiranoj datoteci. Također je i za svaku datoteku dan *TitaniumCore* izvještaj u .json formatu. Primjer jednog izvještaja dan je na slici 2.1.

Slika 2.1. Isječak iz jednog *TitaniumCore* izvještaja

Jedan izvještaj predstavlja popis svojstava pripadne izvršive datoteke do kojih se došlo statičkom analizom datoteke, poput njenih dijelova i njihovih veličina. Uz sve to, dane su nam i oznake 0 - 4 za pojedinu datoteku, pri čemu one označavaju je li datoteka nepakirana (oznaka 0) ili ako je pakirana o kojoj se vrsti pakera radilo (1,2,3 ili 4). Naime, pakeri modificiraju izvršne datoteke na razne načine, među kojima su kompresija podataka, enkripcija podataka i ostalih tehnika koje kod čine nečitljivim korisniku ili drugom programu. Koja oznaka pripada kojoj vrsti pakera navedeno je u tablici 2.2.

Oznaka	Nepakirano ili vrsta pakiranja
0	nije pakirana
1	pakirana - *Overlay*
2	pakirana - *Crypter*
3	pakirana - *Compressing packer*
4	pakirana - *Protector*

Tablica 2.2. Značenje oznaka koje je *ReversingLabs* dodijelio datotekama

Navedimo još i podatke o distribuciji klasa 0-4 u danom trening skupu. Ukupno ima 21 962 nepakiranih i 21 822 pakiranih datoteka. Od pakiranih, njih 1 956 je tipa 1, 1 926 je tipa 2, 14 604 je tipa 3 i 3 336 datoteka je tipa 4.

Zbog različitih veličina i složenosti izvještaja suočili smo se s problemom odabira značajki. Proučavali smo literaturu, primjerice [2] i [3] te koristili značajke koje su autori predložili. Osim toga, sami smo detaljno proučavali izvještaje i tako došli do brojnih novih značajki. U tablici 2.3. navodimo neke od njih.

	Značajka	Zašto smo gledali tu značajku?
1.	log_total_minus_entropy_size	Omjer ukupne veličine svih entrija i onih entrija koji imaju entropy tag. Određujemo koliki postotak memorije ima veliku entropiju. Smatramo da će veliki iznos povećati vjerojatnost da je datoteka pakirana.
2.	max_section_entropy	Uz pretpostavku da sadržaj koji je kriptiran ili obfusciran ili sl. izgleda više kao slučajan niz bajtova, tj. ima veliku entropiju, gledamo najveću vrijednost entropije po sectionima datoteke.
3.	log_average_api_list_size	Packirane datoteke ne bi trebale koristiti previše vanjskih funkcija pa mislimo da će broj navedenih funkcija u API listi imati utjecaj na odluku o pakiranosti datoteke.
4.	last_section_discardable	Gledamo sadrži li posljednji section *discardable* flag, jer smatramo da pakirane datoteke obično sadrže neki bitan kod (npr. skriven ili npr. kod za otpakiravanje) na poslijetku datoteke, pa taj zadnji section ne bi trebao biti označen kao *discardable*.
5.	nonstandard_section_names_num	Neka imena za sectione se smatraju standardnima za PE datoteke (npr. .text ili .data, prema službenom *Microsoftovom* popisu kojeg smo koristili). U literaturi smo pronašli popis takvih imena. Brojimo sectione koji imaju ime koje nije na tom popisu ili uopće nemaju ime.
6.	first_section_entropy	S obzirom da bi prvi section datoteke trebao biti početak te datoteke pa bi kao takav trebao sadržavati neki (koristan) sadržaj, smatramo da njegova entropija ne bi trebala biti prevelika (npr. 7) ili premala (npr. 1).
7.	first_section_wec_code	Prvi sectioni, prema literaturi, obično su .text, tj. sadrže kod, te se smatra da prvi section ne bi trebao imati *write* tag. Stoga gledamo sadrži li prvi section uz *executable* (i naravno *read*) i *write* tag.
8.	average_2_section_entropy	Gledamo težinsku kvadratnu sredinu entropija svih sectiona uzimajući u obzir veličinu pojedinog sectiona. Cilj nam je smanjiti utjecaj malih sectiona s niskim entropijama na srednju vrijednost.
9.	log_max_section_suspiciousness	Iz literature smo zaključili da su veliki sectioni s malom entropijom sumnjivi jer bi u sebi mogli sadržavati skrivene podatke između puno nula što smanjuje entropiju. Za svaki file gledamo max($\frac{velicina\ sectiona}{entropija^2}$) po svim sectionima. Ako se pojavi veliki section sa malom entropijom ovaj će maksimum biti velik
10.	log_num_tags_entropy	Uočili smo vezu između broja entropy tagova i pakiranosti datoteke. Veći broj entropy tagova povećava vjerojatnost da je datoteka pakirana.
11.	diff_log_initialized_log_uninitialized	Zanima nas omjer veličina inicijaliziranih i neinicijaliziranih podataka. Razlika logaritama će nam otprilike dati omjer (neće točno jer dodajemo 1 logaritmu).

Tablica 2.3. Opis nekih važnijih značajki

Algoritam strojnog učenja

Kod odabira modela koristili smo *Weku* te u njoj isprobavali razne modele. Pokrenuli smo i *AutoWeku* kako bismo iz nje dobili čim bolji model. Najboljim klasifikatorom pokazale su se slučajne šume. U nastavku smo koristili njihovu implementaciju *RandomForestClassifier* iz *Pythonove* biblioteke *scikit-learn*.

Zatim je bilo potrebno naći optimalne hiperparametre za što bolje rezultate. Za pronalaženje parametara puno smo puta pokretali model te koristili *grid search*, točnije funkciju *GridSearchCV* iz *scikit-learna*. Promijenjene i defaultne vrijednosti (za parametre koje smo mijenjali) navedene su u tablici 3.1.

PARAMETAR	IZNOS	DEFAULT
n_estimators	800	10
criterion	entropy	gini
min_samples_split	4	2
bootstrap	False	True
n_jobs	-1	1
random_state	42	random

Tablica 3.1. Parametri i njihove vrijednosti

U tablici 3.2. opisujemo njihovo značenje.

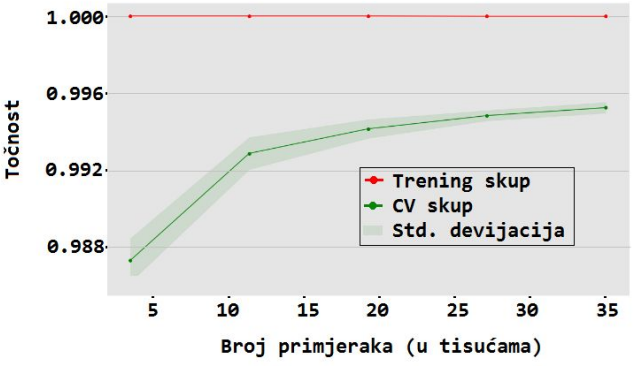
Parametar	**Značenje**
n_estimators	broj stabala koje će slučajna šuma graditi
criterion	funkcija koja mjeri kvalitetu grananja stabala
min_samples_split	minimalni broj uzoraka potrebnih za grananje unutarnjeg čvora stabla
bootstrap	ako je True onda se uzorci u pojedinom stablu mogu ponavljati, inače se može uzeti najviše jedan po pojedinom stablu
n_jobs	broj paralelnih procesa. Ako je -1 onda se vrši maksimalan broj koji procesor može izvršiti
random_state	sjeme za generator pseudoslučajnih brojeva. Nema bitan utjecaj na sam algoritam

Tablica 3.2. Objašnjenje parametara

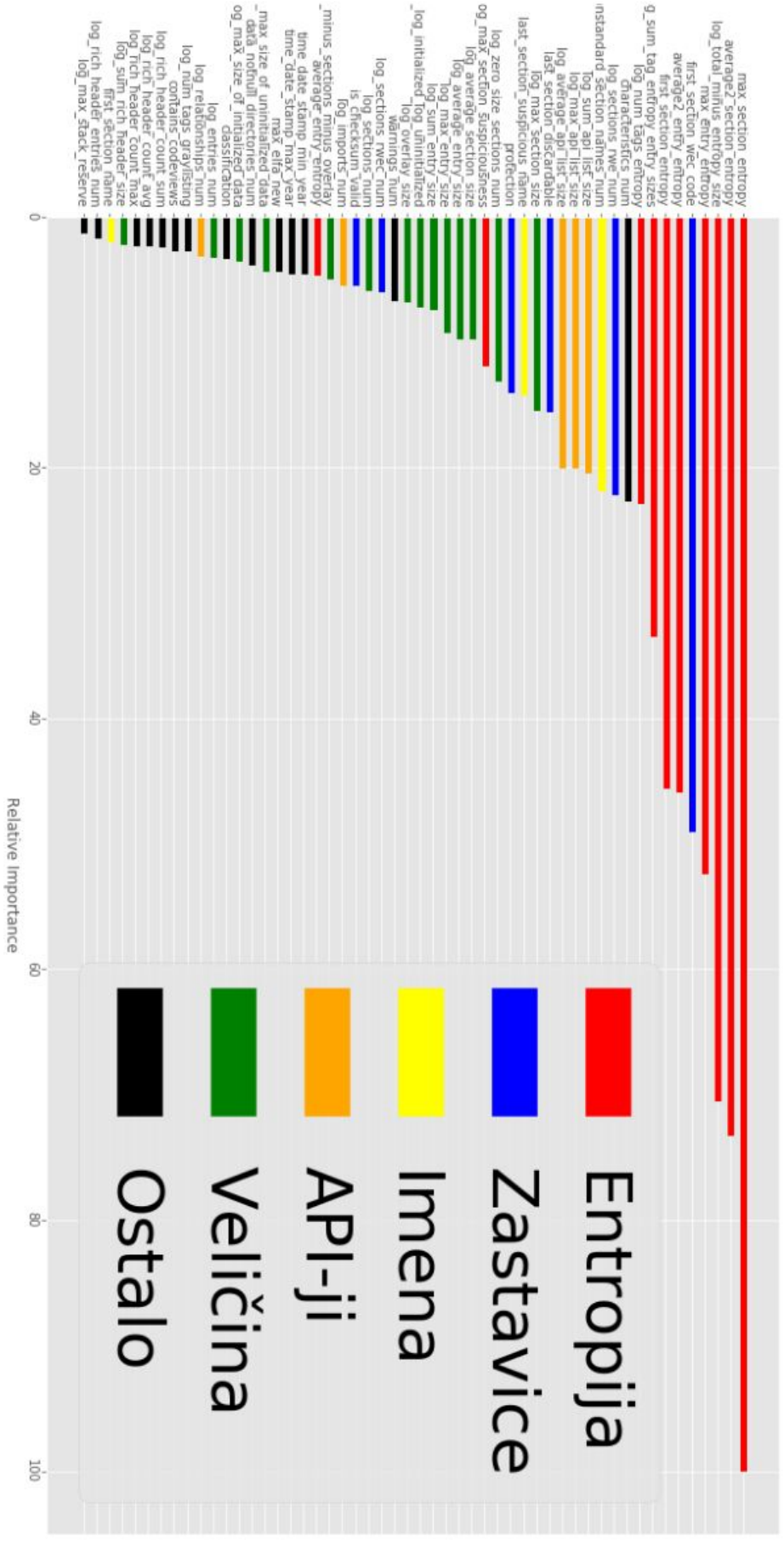
Rezultati

Za provjeru točnosti koristili smo 10-struku unakrsnu validaciju. Iz krivulje učenja na slici 4.1. možemo vidjeti da smo postigli točnost od gotovo 99.5% uz vrlo malu devijaciju točnosti. Također vidimo da ne bismo trebali imati problema sa biasom i varijancom (underfittingom i overfittingom).

U tablici 4.3. je prikazana matrica konfuzije dobivena podjelom na trening i test skup u omjeru 2:1. U testnom skupu se ukupno nalazi 14887 primjeraka.



Slika 4.1. Krivulja učenja



Slika 4.2.
Atributi i njihove važnosti

Na slici 4.2. su prikazane važnosti pojedinih atributa koje nam je dao sam model. Vidimo da su se najvažnijima pokazale značajke vezane za entropiju, zatim značajke vezane za zastavice, imena sekcija i veličine pojedinih dijelova datoteke.

U zelenim se poljima tablice 4.3. nalaze ispravno klasificirani primjerci, a u crvenima neispravno klasificirani. Vidimo da je najteže prepoznavanje klase 1. Općenito su false negativi (kojih je ovdje 48) gori od false positiva (kojih je ovdje 20). Naime, ako pakiranu datoteku proglasimo nepakiranom tada se na njoj neće pokretati univerzalni otpakirač nego će se ona proslijediti antivirusnom programu koji je vjerojatno neće prepoznati.

Matrica konfuzije		predviđeno modelom			
		nepakirano	pakirano	osjetljivost po klasama	ukupna osjetljivost pakiranih
Stvarna klasa	0	7394	20	99,73024%	
	1	34	593	94,57735%	99,35769%
	2	8	676	98,83041%	
	3	5	4990	99,89990%	
	4	1	1166	99,91431%	
	preciznost	99,35501%	99,73136%		
	točnost	99,54323%			

Tablica 4.3. Matrica konfuzije na test skupu

Recimo nešto i o vremenu izvršavanja. Izvlačenje podataka iz datoteka je trajalo cca 12 minuta (iz 43784 datoteka), trening je trajao cca 1 minutu, a predikcija je trajala cca 5 sekundi na 14887 primjeraka.

Rezultati natjecanja Mozgalo 2018.

U sklopu natjecanja Mozgalo dobili smo novi neoznačeni test skup koji se sastojao od ukupno 6928 primjeraka. Među njima se nalazilo i 556 pakiranih datoteka koristeći dosad neviđene pakere. Na tom test skupu smo napravili predikciju koristeći model naučen na početno zadanih 43784 primjeraka. Nakon što smo od organizatora natjecanja dobili oznake, prikazali smo rezultate u tablici 5.1. Vidimo da smo zadržali poprilično veliku točnost. Suprotno očekivanju, u tom se skupu klasa 2 pokazala najtežom za klasifikaciju.

Matrica konfuzije		predviđeno modelom			
		nepakirano	pakirano	osjetljivost po klasama	ukupna osjetljivost pakiranih
Stvarna klasa	0	3399	63	98,18024%	
	1	6	479	98,76289%	97,77842%
	2	46	625	93,14456%	
	3	3	1147	99,73913%	
	4	22	1138	98,10345%	
	preciznost	97,78481%	98,17497%		
	točnost	97,97921%			

Tablica 5.1. Matrica konfuzije na Mozgalovom test skupu

Među 556 primjeraka sa novim pakerima imali smo 536 točno klasificiranih. Stoga zaključujemo da naš algoritam ima dobru sposobnost generalizacije.

U natjecanju su se bodovale: točnost, osjetljivost na pojedinim klasama i na novim pakerima, dokumentacija, analiza rješenja, inovativnost, izvedivost i prezentacija. Konačno, osvojili smo 2. nagradu.

Osvrt na druge pristupe

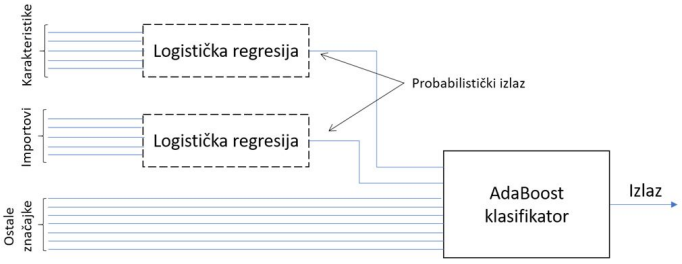
Komentirajmo sada pristupe problemu koji se mogu naći u literaturi. Alat za analizu *Bintropy* otkriva pakiranje analizom samo entropije bajtova. Veliki nedostatak je korištenje samo atributa vezanih za entropiju datoteke. Kolter i suradnici [4] pak koriste n-gram analizu kojom razdvajaju viruse i benigne izvršive datoteke. Nedostatak im je loše raspoznavanje pakiranih od nepakiranih datoteka.

Perdisci i suradnici [2] u svom radu iz 2008. godine, osim entropije mjere za svaku datoteku još i broj standarnih i nestandardnih imena sekcija, broj izvršivih sekcija, broj čitaj/piši/izvrši sekcija, te broj vanjskih funkcija koje datoteka poziva. U svom radu koristili su *Weku*. Preciznije, koristili su naivni Bayesov klasifikator, J48 stablo odlučivanja, ansambl J48 stabala k-najbližih susjeda (sa k=3), višeslojni perceptron (MLP) klasifikator, te *entropy treshold* klasifikator. Najbolje rezultate na svom testnom skupu postigli su sa MLP kojim je dobivena točnost od 98.91% (odnosno, 98.91% datoteka iz testnog skupa je ispravno klasificirano). Isti autori stvorili su i alat za detekciju zloćudnih datoteka *McBoost* koji kombinira heuristike i n-gram analizu bajtova, te koristi stabla odlučivanja i višeslojne perceptrone.

Wang i Wu [3] u svom radu iz 2011. godine koriste stroj s potpornim vektorima (SVM), te (nakon određivanja najboljih parametara) na raznim transformacijama podataka postižu vrlo visoke točnosti. Značajke (uz dodatno profinjavanje) koje koriste su: početna adresa, postoji li nepoznato polje u zaglavlju, broj čitaj/piši/izvrši sekcija, broj vanjskih funkcija, te entropija pojedine sekcije. U treningu je postignuta točnost od 99.93%, dok je u predviđanju postignuta osjetljivost od 94.54%.

Recimo za kraj i nešto o pristupima koje su koristili drugi timovi na natjecanju Mozgalo 2018. Ovdje komentiramo pristupe preostala četiri tima koji su se plasirali u samu završnicu natjecanja. Više detalja o natjecanju može se pronaći na stranici natjecanja Mozgalo [5], a o zadatku u [6]. Detaljnije o pristupima timova nalazi se u prezentaciji [7].

Timovi se također znatno bavili proučavanjem izvještaja, te dijelova datoteke. Tim 505 je korištenjem AdaBoosta i nekoliko značajki koje smo i mi odabrali, postigao točnost od 97% na 43784 primjeraka. Taj tim je također postigao najgore rezultate u usporedbi sa ostalim timovima. Jedan od mogućih razloga za to je poprilično mali broj korištenih značajki. Kao ilustraciju za rad tog tima dajemo dijagram na slici 6.1.



Slika 6.2. Tim 505 koristio je AdaBoost klasifikator

Sljedeći tim *Analysis Paralysis* koristio je XGBOOST (Extreme gradient boosting), pritom se baveći odabirom sljedećih hiperparametara: broj rundi i maksimalna dubina stabala. Ukupno su iz izvještaja izvukli 17 značajki (od kojih se niti jedna posebno ne ističe u odnosu na naše značajke). Oni navode da su probali i logističku regresiju i neuronske mreže, ali su dobili znatno lošije rezultate (točnost oko 95%). XBOOST-om je dobivena točnost od 99.2%, premda je na konačnom testnom skupu točnost ispala manja od naše (što se može vidjeti u tablici 6.2).

Naziv tima	Točnost (%)	FP+FN
Laganica	98,239	122
NewTeam (mi)	97,979	140
Analysis Paraysis	97,546	170
GMO Lazo	97,373	182
505	95,901	284

Tablica 6.2 Točnost koju je pojedini tim ostvario na Mozgalovom test skupu veličine 6928, te broj krivo klasificiranih datoteka (*False positives + false negatives)*

Tim GMO Lazo (što je anagram od mozgalo) koristio je slučajne šume, nakon što je izvukao 60 značajki iz izvještaja, ali i iz datoteka (po čemu su bili inovativniji od ostalih timova).

Iz tablice 6.2. možemo iščitati da je tim Laganica ostvario najveću točnost. Međutim, nemamo detaljnijih informacija o tome kako su to postigli, iako možemo (prema [7]) reći da nema značajnijih razlika u odnosu na naš pristup.

Mogući budući nastavak istraživanja

Postavlja se pitanje kako poboljšati trenutni algoritam. Svakako bismo mogli dodatno proučiti *TitaniumCore* izvještaje i usredotočiti se na detektiranje klasa 1 i 2 koje su se pokazale najtežima za detekciju. Osim toga, mogli bismo pokušati izvući atribute iz samih izvršivih datoteka, po uzoru na tim GMO Lazo.

Literatura

[1] - T. Brosch, M. Morgenstern, *Runtime Packers: The Hidden Problem*, Proc. Black Hat USA, Black Hat,, 2006.

[2] - R. Perdisci, L. Andrea, L. Wenke, *Classification of packed executables for accurate computer virus detection*, Pattern recognition letters, vol. 29, br. 14, str. 1941-1946, 2008.

[3] - W. Tzu-Yen and C. Wu, *Detection of packed executables using support vector machines*, International Conference on Machine Learning and Cybernetics (ICMLC), Vol. 2, IEEE, 2011.

[4] - Kolter J. Z., Maloof M. A., *Learning to Detect and Classify Malicious Executables in the Wild*, Journal of Machine Learning Research 7 (2006)

[5] - Mozgalo, službene stranice: https://www.estudent.hr/category/natjecanja/mozgalo/

[6] - Mozgalo: *Detekcija pakiranih datoteka*, tekst zadatka, dostupno na: https://www.estudent.hr/wp-content/uploads/2018/03/ReversingLabs-zadatak.pdf

[7] - Mozgalo: prezentacija na završnici natjecanja 2018., dostupno na: https://www.estudent.hr/category/natjecanja/mozgalo/#radionice-i-tutoriali_tab