INSTITUTIONEN FÖR REGLERTEKNIK | LUNDS TEKNISKA HÖGSKOLA | PRESENTERAD 2024-01-24

EXAMENSARBETE Towards Automated Log Message Embeddings for Anomaly Detection
STUDENTER Adrian Murphy, Daniel Larsson
HANDLEDARE Johan Eker (LTH), Ola Angelsmark (Advenica)
EXAMINATOR Karl-Erik Årzén (LTH)

Vad säger egentligen ett datorprogram? En studie i att tolka systemloggar

POPULÄRVETENSKAPLIG SAMMANFATTNING **Adrian Murphy, Daniel Larsson**

I det digitaliserade samhället, där samhällskritiska funktioner såsom kollektivtrafik och sjukvård är beroende av IT-infrastruktur, är det viktigare än någonsin att upptäcka systemfel och cyberangrepp i tid. Analys av så kallade loggmeddelanden kan vara ett värdefullt redskap i den kampen. Detta examensarbete utvärderar en ny metod för att tolka loggmeddelanden och jämför den med ett traditionellt tillvägagångssätt.

I berättelsen om Hans och Greta strör barnen brödsmulor längs med vägen för att kunna hitta tillbaka hem. Programmerare använder så kallade loggmeddelanden på ungefär samma sätt. Genom att skriva ut meddelanden på utvalda ställen i programkoden kan man spåra ett IT-systems exekveringsflöde och på så sätt hitta tillbaka till de delar av koden som är av intresse. Ett konkret exempel på hur ett loggmeddelande kan se ut finns i Figur 1 nedan. Avvikelser i den ordning loggmeddelanden skrivs ut eller i det kvantitativa förhållandet mellan olika kategorier av loggmeddelanden kan indikera ett förestående systemfel eller ett pågående cyberangrepp. Genom att dessa avvikelser upptäcks i tid kan man som systemadministratör förbättra sina möjligheter att lösa och avvärja ett eventuellt systemfel eller angrepp.

```
>> 2008-11-09 20:46:55 INFO (...)
   Received block blk_10 of size 6710
```

Figur 1: Ett exempel på hur ett loggmeddelande kan se ut

Eftersom moderna, komplexa IT-system kan generera över 150 miljoner loggmeddelanden i timmen (!) är det omöjligt att upptäcka eventuella avvikelser manuellt. Att upptäcka avvikelser maskinellt, med hjälp av maskininlärningsmetoder, har därför blivit ett populärt forskningsområde. Processen för att upptäcka avvikelser i loggmeddelanden, som ofta kallas för *log anomaly detection* på engelska, kan delas upp i tre steg: (1) tolkning av loggmeddelanden, (2) omvandling till en numerisk representation och slutligen (3) själva anomali- eller avvikelsedetekteringen. I detta arbete undersöks hur en nypåkommen metod för att tolka loggmeddelanden påverkar förmågan att senare upptäcka avvikelser jämfört med ett traditionellt tillvägagångssätt.

Ett traditionellt tillvägagångssätt för att tolka loggmeddelanden kallas Drain. Drain försöker dela in meddelanden i olika kategorier med hjälp av ett så kallat parseträd. Parseträdet tar först hänsyn till antalet ord i ett loggmeddelande och tittar därefter på de enskilda orden som ingår i meddelandet. Genom att dela in loggmeddelanden i olika kategorier blir problemet med att hitta avvikelser mycket mer lätthanterligt, eftersom

antalet kategorier en metod såsom Drain hittar är mycket mindre än det totala antalet loggmeddelanden.

En ny metod för att tolka loggmeddelanden, som vi utvärderar i vårt arbete, är så kallade *logg-embeddings*. Precis som med Drain är ambitionen att dela in meddelanden i olika kategorier. Tillvägagångssättet skiljer sig däremot åt. I stället för att använda ett parseträd använder *logg-embeddings* representationer av enskilda ord som punkter i ett stort matematiskt rum. Dessa enskilda punkter, där ord som är nära besläktade (till exempel skicka och sända) ligger nära varandra i det matematiska rummet, slås i sin tur samman till en enda punkt för hela loggmeddelandet. Loggpunkterna kan i sin tur grupperas i olika kategorier, där meddelanden som är semantiskt lika, trots att de kanske inte innehåller exakt samma ord, kan slås samman till en och samma kategori.

En utmaning med att tolka loggmeddelanden är att formatet kan förändras över tid. En programmerare kan till exempel byta ut, lägga till eller ta bort enstaka ord i loggmeddelandet. En programmerare kan dessutom kan lägga till en helt ny typ av loggmeddelande i koden. Inom logganalys kallas detta fenomen för *concept drift* på engelska. I vårt arbete har vi undersökt hur väl Drain och *logg-embeddings* tål *concept drift*, alltså hur själva förmågan att upptäcka avvikelser påverkas i ett senare steg, genom att byta ut enstaka ord i loggmeddelanden mot synonymer.

Våra resultat visar på att förmågan att upptäcka avvikelser i systemloggar i ett scenario utan *concept drift* inte påverkas av huruvida man använder Drain eller *logg-embeddings* som tolkningsmetod. I ett scenario med *concept drift* försämras däremot förmågan att upptäcka avvikelser markant med Drain, samtidigt som *logg-embeddings* nästan inte påverkas alls.

Sammanfattningsvis styrker vårt arbete att *logg-embeddings* utgör en lovande tolkningsmetod för att upptäcka avvikelser i systemloggar, särskilt i situationer med *concept drift*. Medan traditionella metoder som Drain påverkas negativt av förändringar i loggformatet, visar *logg-embeddings* på en imponerande robusthet. Denna metod öppnar dörren för effektiv logganalys i moderna, dynamiska IT-miljöer där loggvolymen är enorm och förändringar är oundvikliga.